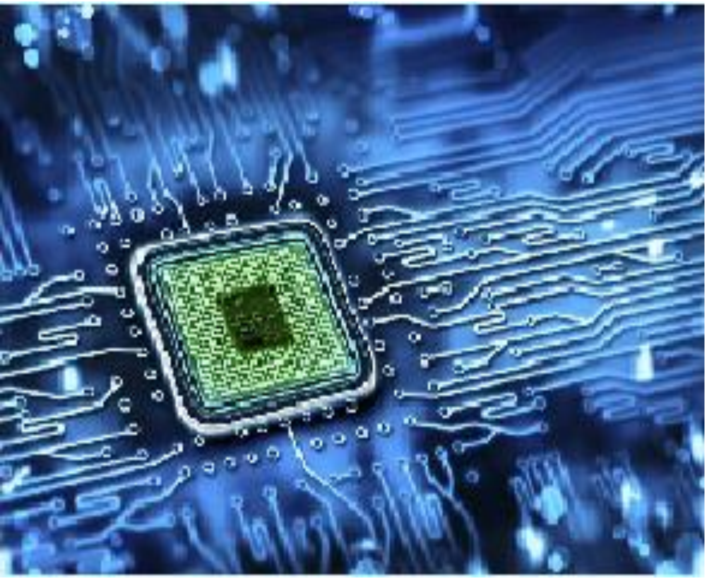
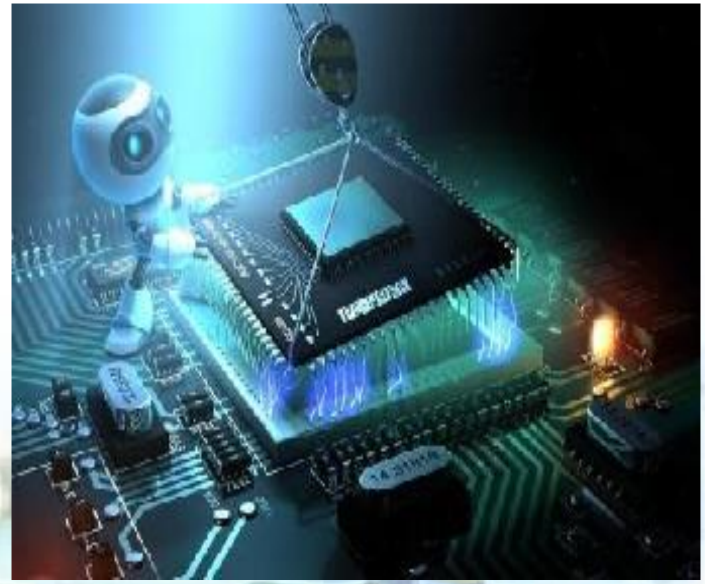
16CS402 - Embedded Systems

VIGNAN'S
Foundation for Science, Technology & Research
(Deemed to be UNIVERSITY)
-Estd. u/s 3 of UGC Act 1956

4th Year 1st Semester



CSE



Dr. V. VIJAYARAGHAVAN, M.E., Ph.D.,

Assistant Professor - ECE, VFSTR, Guntur, AP.

UNIT - I

Introduction to Embedded Systems

- Definition, Applications of Embedded Systems
- Embedded Hardware Units and Devices
- Embedded Software
- Design Metrics in ES, Challenges in ES design
- Host and Target Machines Getting ES in to Target System

UNIT – I INTRODUCTION TO EMBEDDED SYSTEMS

Definition:

An embedded system is a system that has embedded software in a hardware and it is dedicated for either an application (or) specific part of an application but not it self intended to be a general purpose computer.

Ex:- Washing Machine, DVD Player, Digital Camera, Air Conditioner, Printer, Mobile phones, etc.,

An embedded system has three main components:

- ➢ It has hardware.
- ➢ It has application software.
- ➢ It has Real Time Operating system (RTOS)

(The embedded software is also called 'firmware')

Definition……

Characteristics:

- Real-time and Multi-rate operations,
- Complex algorithms,
- Complex GUIs and other user interfaces,
- Dedicated functions.

Constraints:

- Available system memory,
- Available processor speed,
- Limits power dissipation.

(Performance, Power, Size and Cost are other design constraints).

Definition……

EMBEDDED SYSTEM & GENERAL PURPOSE COMPUTER:

GENERAL PURPOSE COMPUTER	EMBEDDED SYSTEM
It is combination of generic hardware and a general purpose OS for executing a variety of applications.	It is combination of special purpose hardware and embedded OS for executing specific set of applications.
It contains general purpose operating system.	It may or may not contain operating system.
Applications are alterable by the user.	Applications are non-alterable by the user.
Performance is the key factor.	Application specific requirements are the key factors.
More Power Consumption.	Less Power Consumption.
Response Time is Not Critical.	Response Time is Critical for some applications.

Definition……

Categories of Embedded Systems:

Embedded systems are classified into four categories based on their performance and functional requirements:

- Stand-alone Embedded Systems
- Real-time Embedded Systems
- Networked embedded systems
- Mobile embedded systems.

Embedded Systems are classified into three types based on the performance of the microcontroller:

- Small scale embedded systems
- Medium scale embedded systems
- Sophisticated embedded systems.

Categories:

i) Stand-alone Embedded Systems

Stand-alone systems take inputs, process them and produce the desired output.

The input can be electrical signals from transducers or commands from a human being. 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.

Ex: Digital Camera, Microwave Oven, CD Player, Air conditioner, TV etc.

Categories:
ii) 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**.

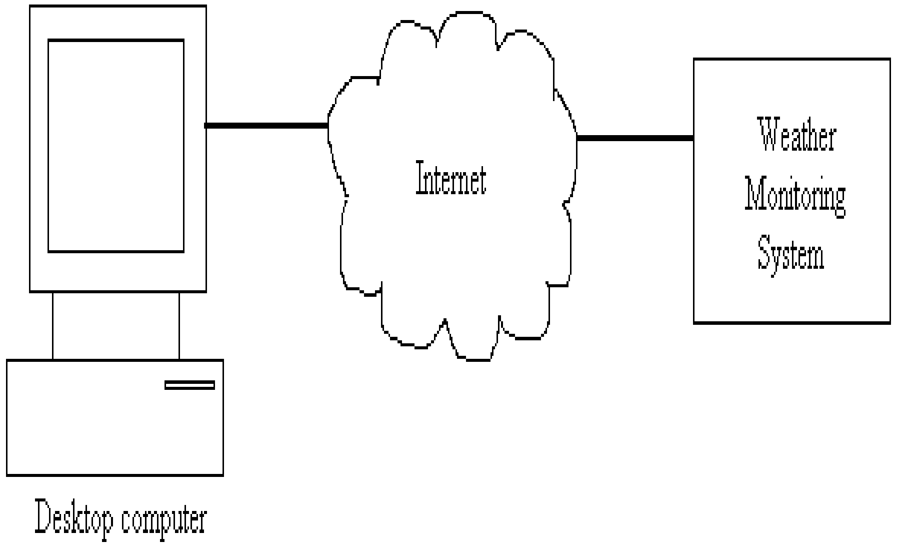
In some embedded systems, deadlines are imposed, but not adhering to them once in a while may not lead to a catastrophe. For example, you give a command to the DVD player from a remote control, and there is a delay of a few milliseconds in executing that command. But, this delay won't lead to a serious implication. Such systems are called **soft real-time systems**.

Categories:

iii) 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.
TCP/IP (Transmission Control Protocol/Internet Protocol) protocol suite is used.

Figure 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.



Categories:

iv) 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.

Based on the Performance of the Microcontroller

i) Small Scale Embedded Systems

These types of embedded systems are designed with a single 8 or 16-bit microcontroller. For developing embedded software, the main programming tools are an editor, assembler, cross assembler and integrated development environment (IDE).

ii) Medium Scale Embedded Systems

These types of embedded systems design with a single or 16 or 32 bit microcontroller, RISCs or DSPs. For developing embedded software, the main programming tools are C, C++, JAVA, Visual C++, RTOS, debugger, source code engineering tool, simulator and IDE.

iii) Sophisticated Embedded Systems

These types of embedded systems have enormous hardware and software complexities, that may need ASIPs, IPs, PLAs, scalable or configurable processors.

UNIT – I INTRODUCTION TO EMBEDDED SYSTEMS

Applications:

Embedded systems are used in different applications like automobiles, telecommunications, smart cards, missiles, satellites, computer networking and digital consumer electronics.

Embedded Systems in Automobiles and in telecommunications

- Motor and cruise control system
- Body or Engine safety
- Entertainment and multimedia in car
- E-Com and Mobile access
- Robotics in assembly line
- Wireless communication
- Mobile computing and networking

Applications…..

Embedded Systems in Smart Cards, Missiles and Satellites

- Security systems
- Telephone and banking
- Defence and aerospace
- Communication

Embedded Systems in Peripherals & Computer Networking

- Displays and Monitors
- Networking Systems
- Image Processing
- Network cards and printers

Embedded Systems in Consumer Electronics

- Digital Cameras
- Set top Boxes
- High Definition TVs
- DVDs

Applications.....

i) Consumer Appliances

Digital camera, digital diary, DVD player, electronic toys, microwave oven, remote controls for TV and air conditioner, VCD player, video game consoles, video recorders etc.

Ex: Digital diary: Casio digital diaries are electronic organizers capable of storing hundreds or thousands of memo, address, and schedule records.

Features of Digital diary:

- Telephone directory
- Schedule keeper
- Memo function
- To do list
- World time
- Secret memory area
- Alarm
- Metric & Currency conversion function

Applications.....

ii) Office Automation

The office automation products using embedded systems are copying machine, fax machine, key telephone, modem, printer, scanner etc.

Ex: Modem :

A modem (modulator-demodulator) is a device that modulates an analog carrier signal to encode digital information, and also demodulates such a carrier signal to decode the transmitted information. The goal is to produce a signal that can be transmitted easily and decoded to reproduce the original digital data.

Modems are generally classified by the amount of data they can send in a given unit of time, usually expressed in bits per second (bit/s, or bps).

iii) Industrial Automation

The embedded systems for industrial use are designed to carry out specific tasks such as monitoring the temperature, pressure, humidity, voltage, current etc.,

Thirty years earlier, the Volkswagen 1600 used a microprocessor to control its fuel injection, making it the first embedded system in the auto industry.

Ex: Industrial Robot:

An industrial robot is as an automatically controlled, reprogrammable, multipurpose manipulator programmable in three or more axes. Typical applications of robots include welding, painting, assembly, pick and place, product inspection, and testing; all accomplished with high speed, and precision.

iv) Medical Electronics

Almost every medical equipment in the hospital is an embedded system. These equipments include diagnostic aids such as ECG, EEG, blood pressure measuring devices, X-ray scanners; equipment used in blood analysis, radiation, endoscopy etc.

- Electrocardiogram (ECG) is used to measure the rate and regularity of heartbeats as well as the size and position of the chambers, the presence of any damage to the heart, and the effects of drugs or devices used to regulate the heart (such as a pacemaker).
- Electroencephalography (EEG) is the recording of electrical activity along the scalp. EEG measures voltage fluctuations resulting from ionic current flows within the neurons of the brain.

Applications.....

v) Computer Networking

Bridges, routers, Integrated Services Digital Networks (ISDN), Asynchronous Transfer Mode (ATM), X.25 and frame relay switches are embedded systems which implement the necessary data communication protocols.

vi) Tele Communications

ISDN phones, terminal adapters, web cameras, packet assembler/disassembler are embedded systems.

vii) Wireless Technologies

The mobile phone is one of the marvels of the last decade of the 20th century. It is a very powerful embedded system that provides voice communication while we are on the move.

The Personal Digital Assistants and the palmtops can now be used to access multimedia services over the Internet.

Applications.....

viii) Instrumentation

Testing and measurement are the fundamental requirements in all scientific and engineering activities. The measuring equipment we use in laboratories to measure parameters such as weight, temperature, pressure, humidity, voltage, current etc are all embedded systems. Test equipment such as oscilloscope, spectrum analyzer, logic analyzer, protocol analyzer, radio communication test set etc. are embedded systems built around powerful processors.

ix) Security

Security of persons and information has always been a major issue. We need to protect our homes and offices; and also the information we transmit and store. Developing embedded systems for security applications is one of the most lucrative business nowadays.

Applications.....

ix) Security...

Security devices at homes, offices, airports etc. for authentication and verification are embedded systems. Encryption devices are used to encrypt the data/voice being transmitted on communication links such as telephone lines. Biometric systems using fingerprints and face recognition are now being extensively used for user authentication in banking applications as well as for access control in high security buildings.

Home security devices:

- ➢ Closed-circuit TV
- ➢ Door and/or window contacts
- ➢ Glass break detector
- ➢ Motion detector
- ➢ Wireless panic button
- ➢ Monitored fire detection system

x) Finance

Financial dealing through cash and cheques are now slowly paving way for transactions using smart cards and ATM (Automatic Teller Machine or Any Time Money) machines.

Credit Card:

A credit card is a small plastic card issued to users as a system of payment. It allows its holder to buy goods and services based on the holder's promise to pay for these goods and services.

Smart Card:

A smart card contain volatile memory and microprocessor components. Smart cards can provide identification, authentication, data storage and application processing.
A single contact/contactless smart card can be programmed with multiple banking credentials, medical entitlement, driver’s license/public transport entitlement, loyalty programs and club memberships, etc.

Applications.....

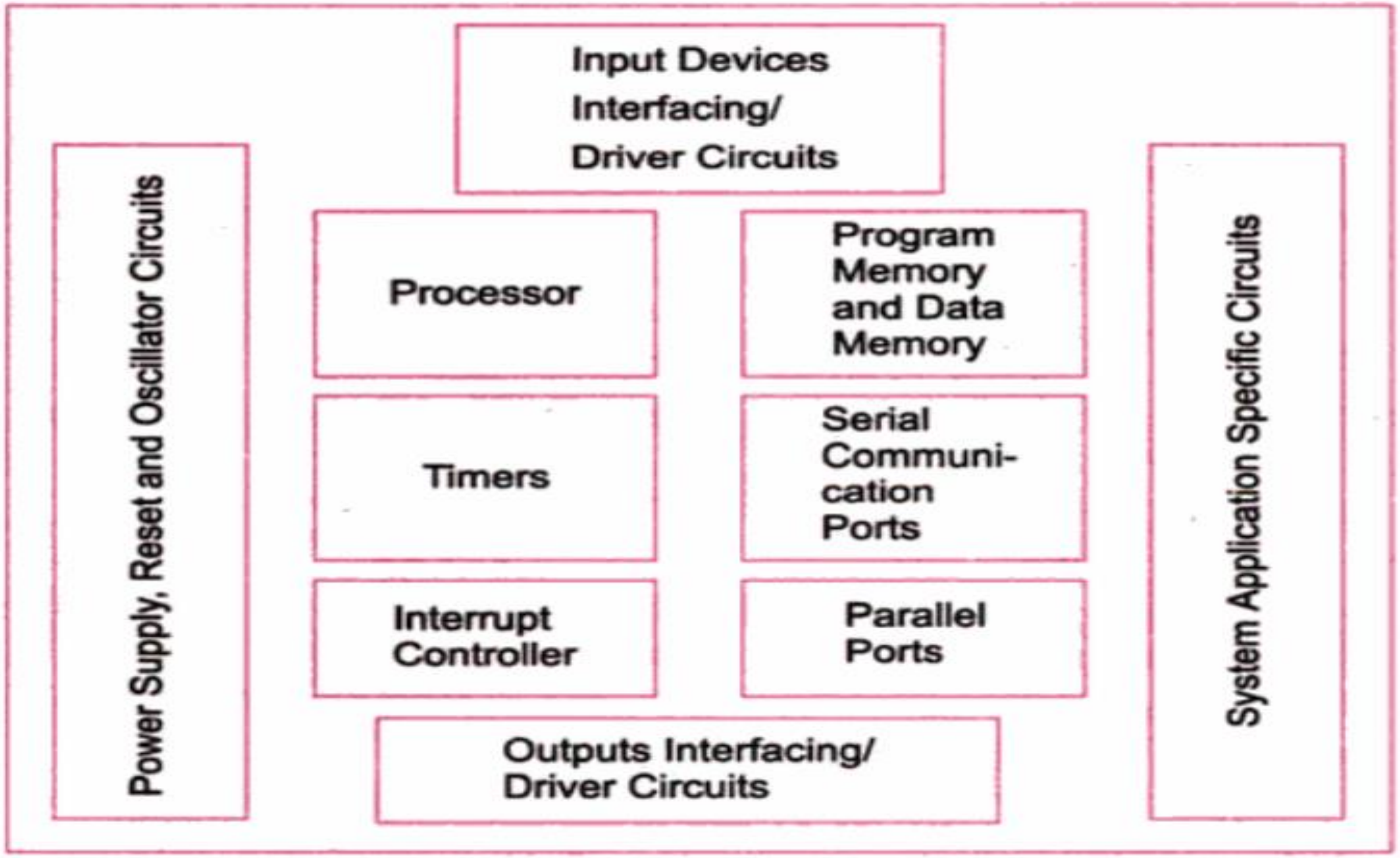
S.No	Embedded System	Application
1	Home Appliances	Dishwasher, washing machine, microwave oven, set-top box, security system, DVD, answering machine, garden sprinkler systems etc..
2	Office Automation	Fax, copy machine, smart phone system, modern, scanner, printers.
3	Security	Face recognition, finger recognition, eye recognition, building security system , airport security system, alarm system.
4	Academia	Smart board, smart room, OCR, calculator, smart cord.
5	Instrumentation	Signal generator, signal processor, power supplier, Process instrumentation,
6	Telecommunication	Router, hub, cellular phone, IP phone, web camera
7	Automobile	Fuel injection controller, anti-locking brake system, air bag system, GPS, cruise control.

Applications.....

S.No	Embedded System	Application
8	Entertainment	MP3, video game, Mind Storm, smart toy.
9	Aerospace	Navigation system, automatic landing system, flight attitude controller, space explorer, space robotics.
10	Industrial automation	Assembly line, data collection system, monitoring systems on pressure, voltage, current, temperature, hazard detecting system, industrial robot.
11	Personal	PDA, iPhone, palmtop, data organizer.
12	Medical	CT scanner, ECG , EEG , EMG ,MRI, Glucose monitor, blood pressure monitor, medical diagnostic device.
13	Banking & Finance	ATM, smart vendor machine, cash register ,Share market
14	Miscellaneous:	Elevators, tread mill, smart card, security door etc.

Embedded Hardware Units and Devices:

Every embedded system consists of custom-built hardware.


Input Devices Interfacing/ Driver Circuits
Power Supply, Reset and Oscillator Circuits
Processor
Program Memory and Data Memory
Timers
Serial Communi-cation Ports
Interrupt Controller
Parallel Ports
Outputs Interfacing/ Driver Circuits
System Application Specific Circuits

Embedded Hardware Units and Devices.....

1. Processor:

Processor has two essential units

- **Program Flow and data path Control Unit** (CU) - includes a fetch unit for fetching instructions from the memory.
- **Execution Unit** (EU) - includes circuits for arithmetic and logical unit (ALU), and for instructions for a program control task, data transfer instructions, halt, interrupt, or jump to another set of instructions or call to another routine or sleep or reset.

Embedded System Processor can be

- General purpose microprocessor
  - Microprocessor - Intel 80x86, Sparc, or Motorola 68HCxxx.
  - Embedded general purpose processor - Intel Xscale for PDAs.
- Application Specific Instruction-Set Processor (ASIP)
  - Microcontroller - Intel MCS51, Philips 51XA, or Motorola 68HC11.
  - DSP or Media processor or Network processor or IO processor.
- Application Specific System Processor (ASSP)
  - Set top box processor or mpeg video-processor or mobile application processor.

Embedded Hardware Units and Devices.....

2. Power Source:

- Most systems have a power supply of own. NICs and Graphic accelerators are do not have their own power supply and connect to processor power-supply lines.
- Operated Power ranges
  - 5V ±0.25V
  - 3.3V ±0.3V
  - 2V ±0.2V
  - 1.5V ±0.2V
- **Charge pump** concept used in a system of little power needs, for examples, in the mouse or contact-less smart card.
  - charge pump consists of a diode in the series followed by a charging capacitor used to accumulate charge from bus signals.

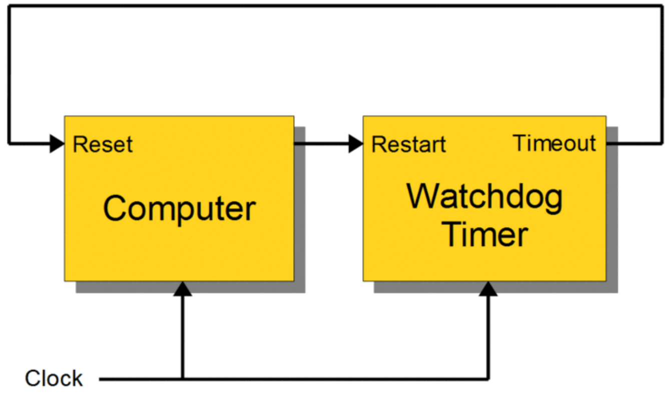
Embedded Hardware Units and Devices…..

3. Clock Oscillator Circuit and Clocking Unit:

- The Clock controls the time for executing an instruction.
- The Clock is an important unit of a system. A processor needs a clock oscillator circuit.
- The clock controls the various clocking requirements of the CPU, of the system timers and the CPU machine cycles.

4. Reset Circuit:

- Reset on Power-up
- External and Internal Reset circuit
- Reset on Timeout of Watchdog timer.



Embedded Hardware Units and Devices…..

5. System timer and Real Time Clock (RTC):

- A timer circuit suitably configured as the **system-clock**, which ticks and generates system interrupts periodically.
- A timer circuit suitably configured as **real-time clock** (RTC). Used by schedulers, time and date of the system and for real-time programming.

6. Interrupt Handler:

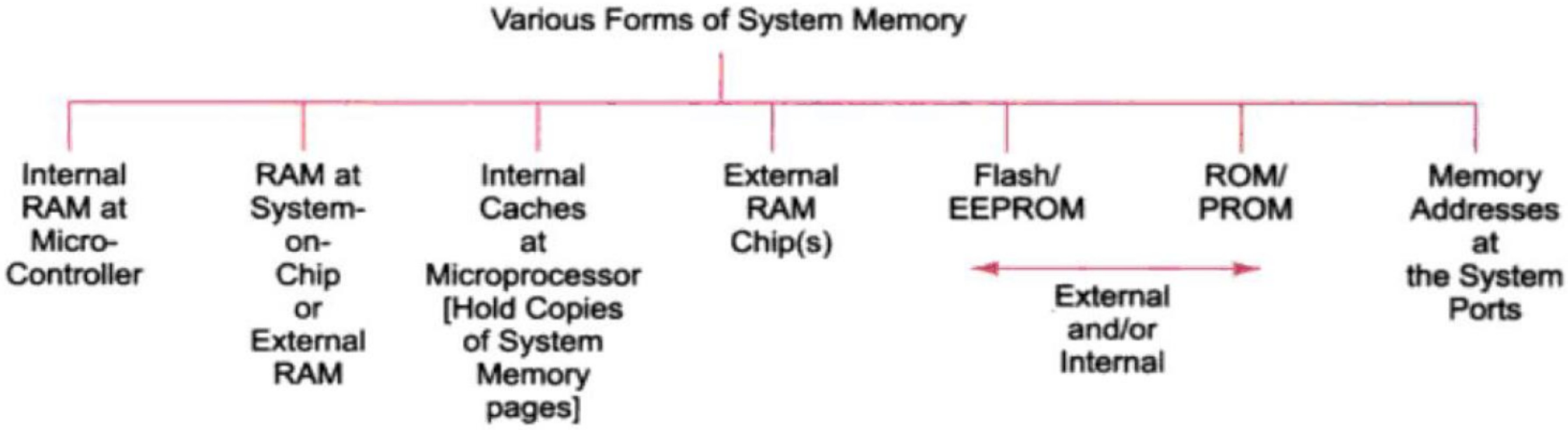
- Interrupt Handling element for the external port interrupts, IO interrupts, timer and RTC interrupts, software interrupts and exceptions.
- System may prioritize the sources and service accordingly.
- Some sources are non-maskable.

Embedded Hardware Units and Devices.....

7. Memory:

In a system, there are various types of memories. They are as follows:

- Internal RAM of 256 or 512 bytes for registers, temporary data stack.
- Internal ROM/PROM/EEPROM for about 4 KB to 16 KB of program.
- External RAM for the temporary data and stack or Internal caches.
- Internal flash.
- Memory sticks.
- External ROM or PROM for embedding software.
- RAM Memory buffers at ports.
- Caches (in superscalar microprocessors).

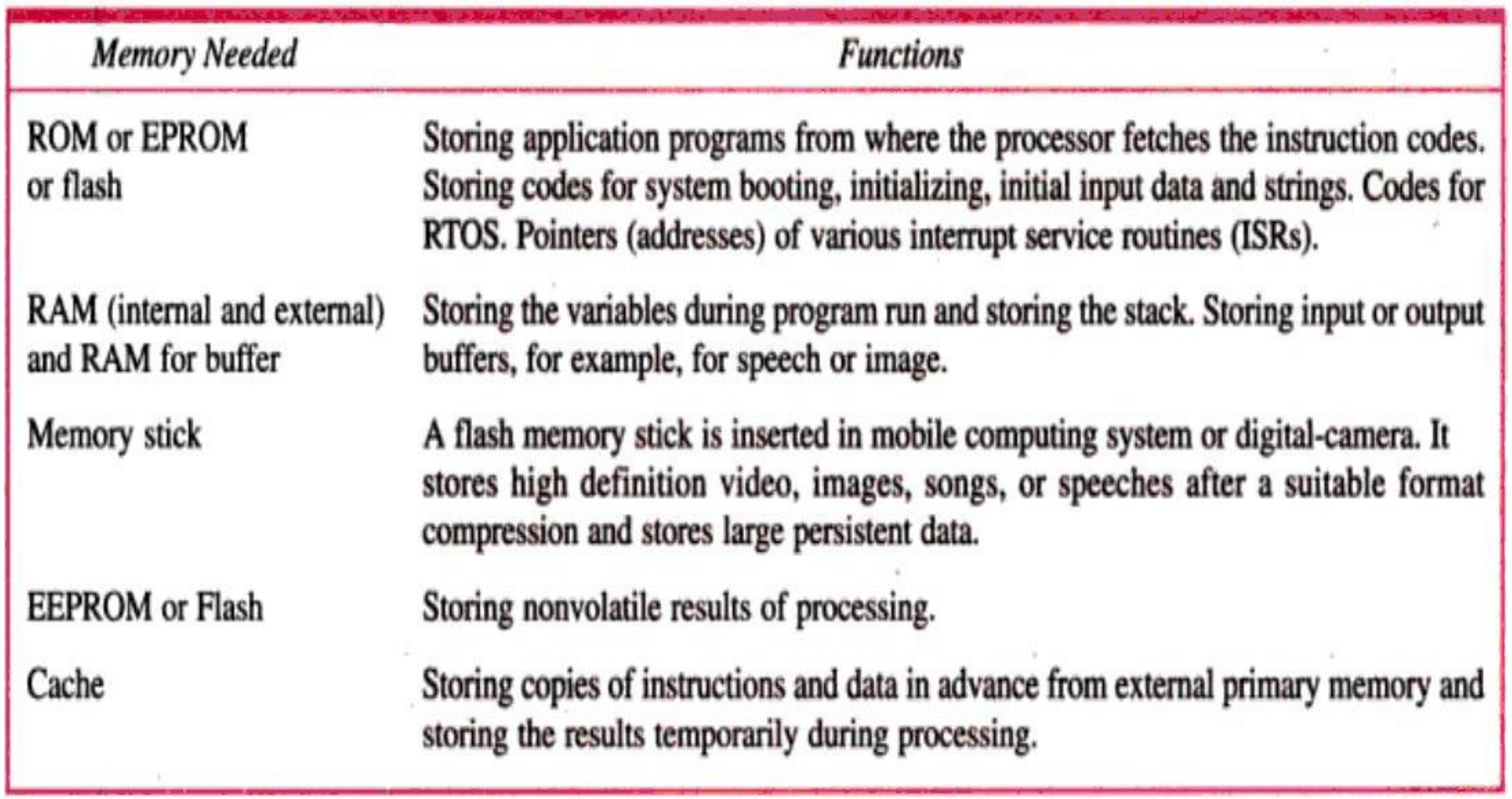


Various Forms of System Memory
Internal RAM at Micro-Controller
RAM at System-on-Chip or External RAM
Internal Caches at Microprocessor [Hold Copies of System Memory pages]
External RAM Chip(s)
Flash/ EEPROM
ROM/ PROM
External and/or Internal
Memory Addresses at the System Ports

Embedded Hardware Units and Devices…..

7. Memory……

Functions assigned to the memories in a system



Memory Needed	*Functions*
ROM or EPROM or flash	Storing application programs from where the processor fetches the instruction codes. Storing codes for system booting, initializing, initial input data and strings. Codes for RTOS. Pointers (addresses) of various interrupt service routines (ISRs).
RAM (internal and external) and RAM for buffer	Storing the variables during program run and storing the stack. Storing input or output buffers, for example, for speech or image.
Memory stick	A flash memory stick is inserted in mobile computing system or digital-camera. It stores high definition video, images, songs, or speeches after a suitable format compression and stores large persistent data.
EEPROM or Flash	Storing nonvolatile results of processing.
Cache	Storing copies of instructions and data in advance from external primary memory and storing the results temporarily during processing.

Embedded Hardware Units and Devices…..

7. Memory…… Memory Comparison Table

Type	Volatile?	Writeable?	Erase Size	Max Erase Cycles	Cost (per Byte)	Speed
SRAM	Yes	Yes	Byte	Unlimited	Expensive	Fast
DRAM	Yes	Yes	Byte	Unlimited	Moderate	Moderate
Masked ROM	No	No	n/a	n/a	Inexpensive	Fast
PROM	No	Once	n/a	n/a	Moderate	Fast
EPROM	No	Yes,	Entire Chip	Limited	Moderate	Fast
EEPROM	No	Yes	Byte	Limited	Expensive	Fast to read, slow to erase/write
Flash	No	Yes	Sector	Limited	Moderate	Fast to read, slow to erase/write
NVRAM	No	Yes	Byte	Unlimited	Expensive (SRAM + battery)	Fast

Embedded Hardware Units and Devices…..

8. Input/Output Devices

- Function Keypad & Touchscreen - to input data and/or commands.
- Light Emitting Diodes (LEDs) - status display or indications of visual alarms for such events as power supply failure.
- Liquid Crystal Displays (LCDs) - to display the status information.
- Microphone and speakers - Microphone converts the acoustic energy into a voltage signal. The speakers convert the electrical signal back into acoustic waves.
- Video camera and monitor - The real life image or scenery is converted into electrical signal using a video camera. The electrical signal is converted back to the image on a monitor.
- Sensors and Transducers - to convert real life information into equivalent electrical signals.
  - Temperature sensors, Light sensors, Accelerometer, Pressure sensors
- ADC – Single or Multi channel and DAC.
- Modem, Bluetooth, 802.11, IrDA….

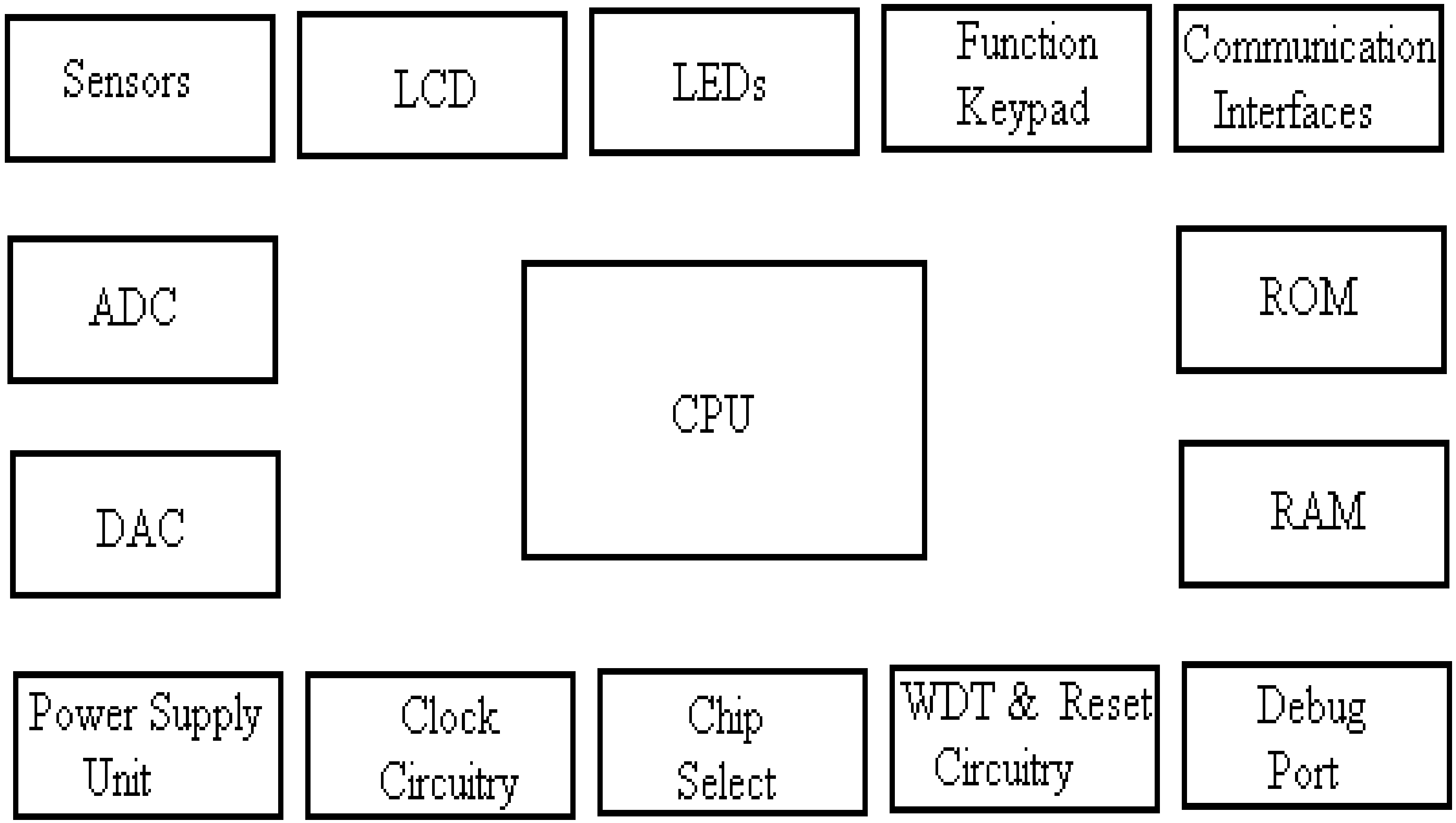
Embedded Hardware Units and Devices…..

9. Linking Embedded System Hardware

- Linking and interfacing circuit for the Buses by using the appropriate multiplexers, and decoders, demultiplexers Interface the various system units.
- Communication Driver: Network Ethernet or serial driver to communicate with host embedded system expansion facility.
- Serial Buses (Example): UART (512 kbaud/s), Serial Port (230 kbps), 1-wire CAN (33 kbps), Fault tolerant CAN (110 kbps), Industrial I2C (100kbps), SPI (100 kbps), MicroWire (300 kbps).
- Parallel Buses (Example): PCI, PCI-X
- SCSI (Small Computer System Interface) parallel (40 Mbps), Fast SCSI (80 Mbps), Ultra SCSI-3 (160 Mbps), FireWire/IEEE 1394 (400 Mbps), High Speed USB 2.0 (480 Mbps).

Hardware Architecture:

The various building block of embedded system hardware are shown in Fig.



UNIT – I INTRODUCTION TO EMBEDDED SYSTEMS

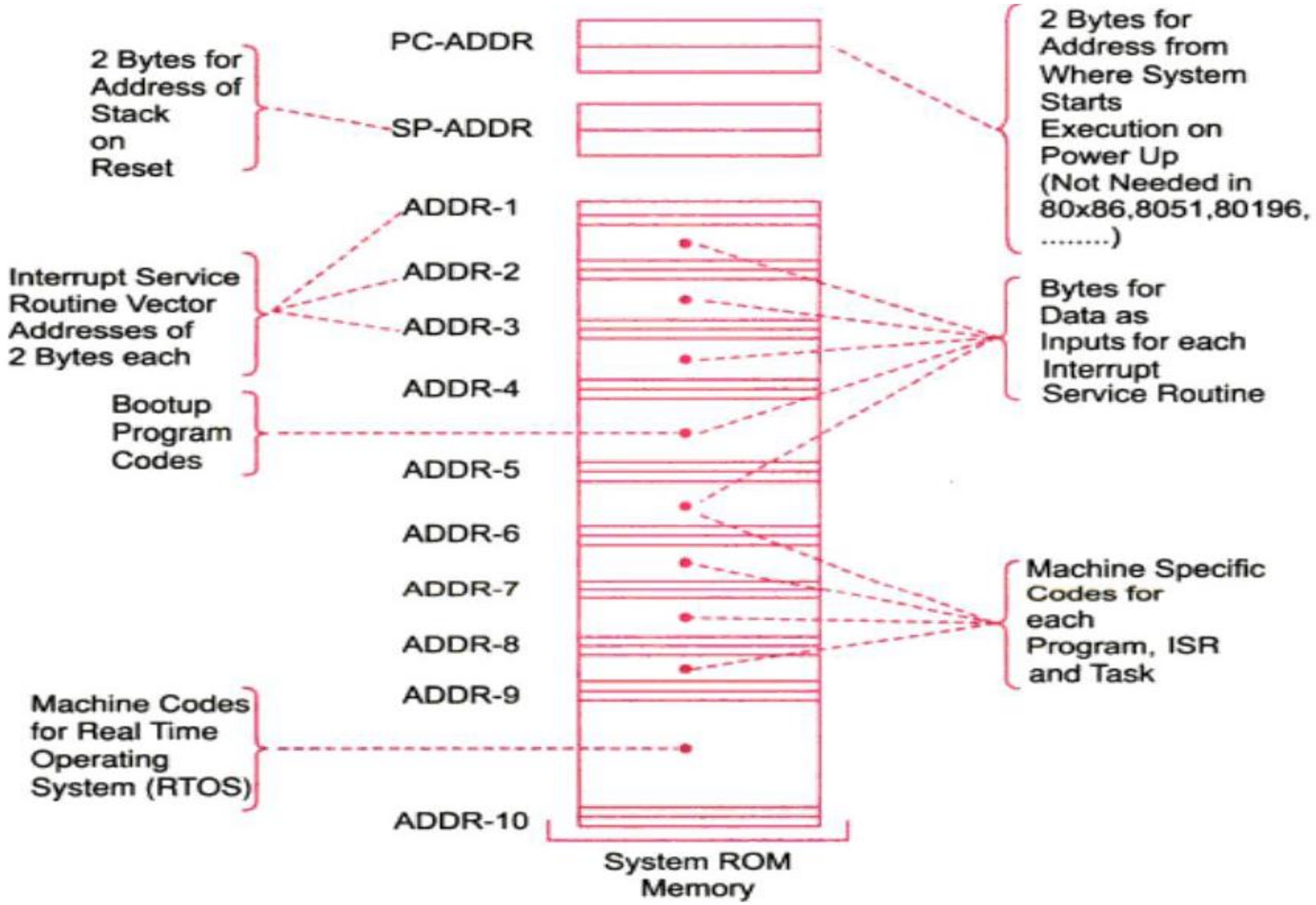
Embedded Software:

The Software is like a brain of the embedded system.

1. Final Machine Implementable Software for a System:

- An Embedded system processor executes software that is specific to a given application of that system.
- The instruction codes and data in the final phase are placed in the ROM or Flash memory. This is also called **ROM image**.
- Bytes at each address defined for creating the ROM image. By changing this image, the same hardware platform work differently and can be used for entirely different applications or for new upgrades of the same system.
- Hardware elements between the distinct systems can be identical but it is the software that makes a system unique and distinct from the other.
- ROM image may alternatively be compressed software (for example, the zip format) and data (for example, the pictures in jpg or gif format) along with the software required for decompression algorithm.

Embedded Hardware Units and Devices…..


PC-ADDR
2 Bytes for Address of Stack on Reset
SP-ADDR
2 Bytes for Address from Where System Starts Execution on Power Up (Not Needed in 80x86,8051,80196, ………)
ADDR-1
ADDR-2
Interrupt Service Routine Vector Addresses of 2 Bytes each
ADDR-3
Bytes for Data as Inputs for each Interrupt Service Routine
ADDR-4
Bootup Program Codes
ADDR-5
ADDR-6
ADDR-7
Machine Specific Codes for each Program, ISR and Task
ADDR-8
ADDR-9
Machine Codes for Real Time Operating System (RTOS)
ADDR-10
System ROM Memory

System ROM memory embedding the software, RTOS, data, and vector addresses

Dr. V.Vijayaraghavan, Assistant Professor-ECE, VFSTR, Guntur

Embedded Hardware Units and Devices.....

2. Coding of Software in Machine Codes:

- Programmer defines the addresses and the corresponding bytes or bits at each address.
- Used in configuring some specific physical device or subsystem like transceiver, the machine code- based coding is used.

3. Software in Processor Specific Assembly Language:

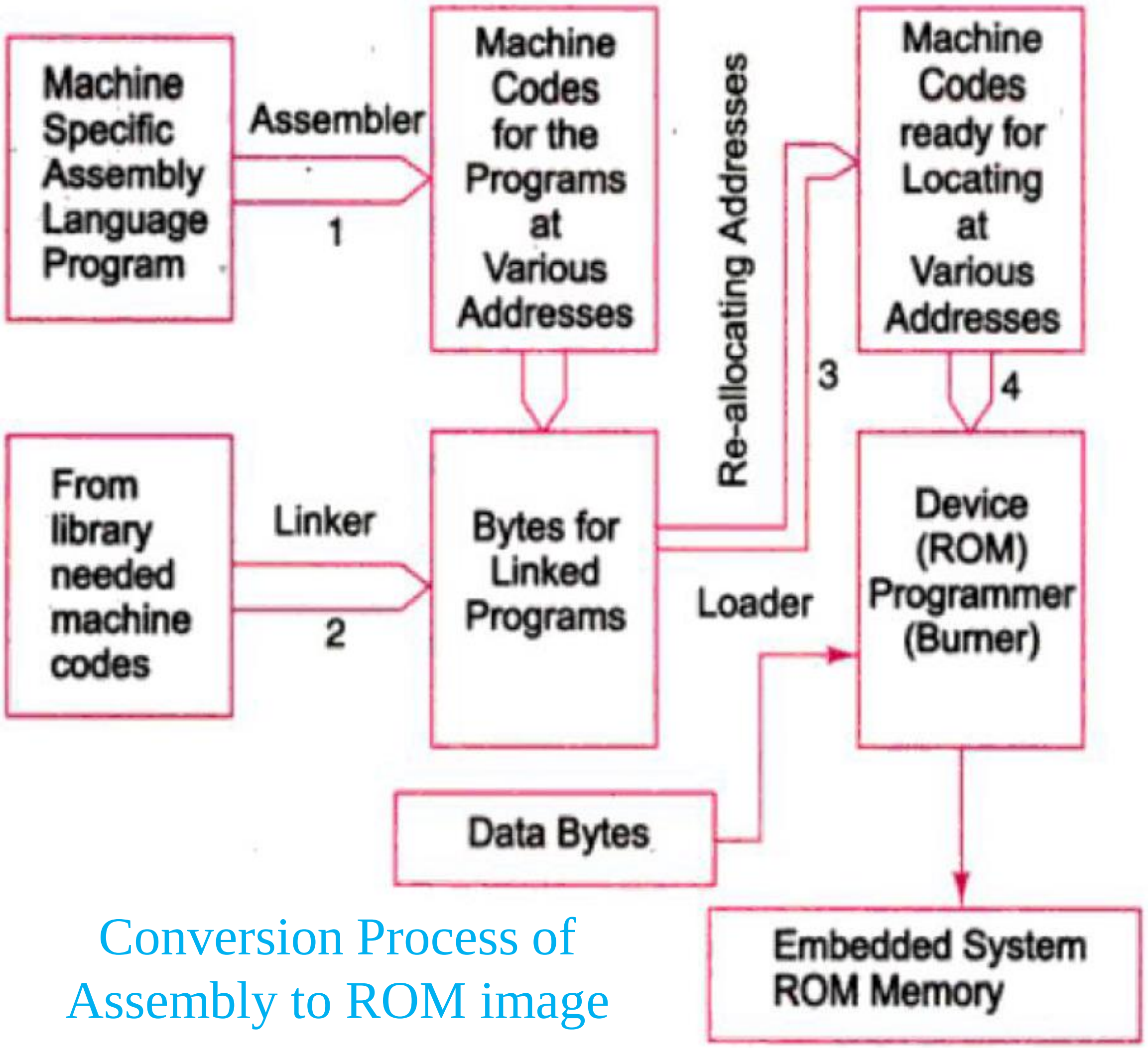
- Needed for Invoking Processor Specific Instructions, Requires understanding of the processor and instruction set.
- A program or a small specific part coded in the assembly language using an Assembler (software used for developing codes in assembly).
- It is extremely useful for configuring physical devices like Ports, Line display, ADC, DAC…etc.
- To make all the codes in assembly is very time consuming.
- Full coding in assembly may be done for simple small scale systems.

Embedded Hardware Units and Devices…..

3. Software in Processor Specific Assembly Language…

The steps followed when using assembly language before finally burned at the ROM

- 'Assembler'- Translates ALP to Machine codes.
- 'Linker' – Link the codes with the other codes. The linked file in binary for run is known as exe file.
- 'Loader' – Relocating the codes after finding the physical memory addresses available.
- 'Locator' – Locating these codes as a ROM image. It locates IO tasks and hardware device drivers codes at fixed addresses. (hex file)
- 'Device Programmer' – Takes ROM image and burns into PROM or Flash.


Machine Specific Assembly Language Program
Assembler
1
Machine Codes for the Programs at Various Addresses
Re-allocating Addresses
3
Machine Codes ready for Locating at Various Addresses
4
From library needed machine codes
Linker
2
Bytes for Linked Programs
Loader
Device (ROM) Programmer (Burner)
Data Bytes
Embedded System ROM Memory

Conversion Process of Assembly to ROM image

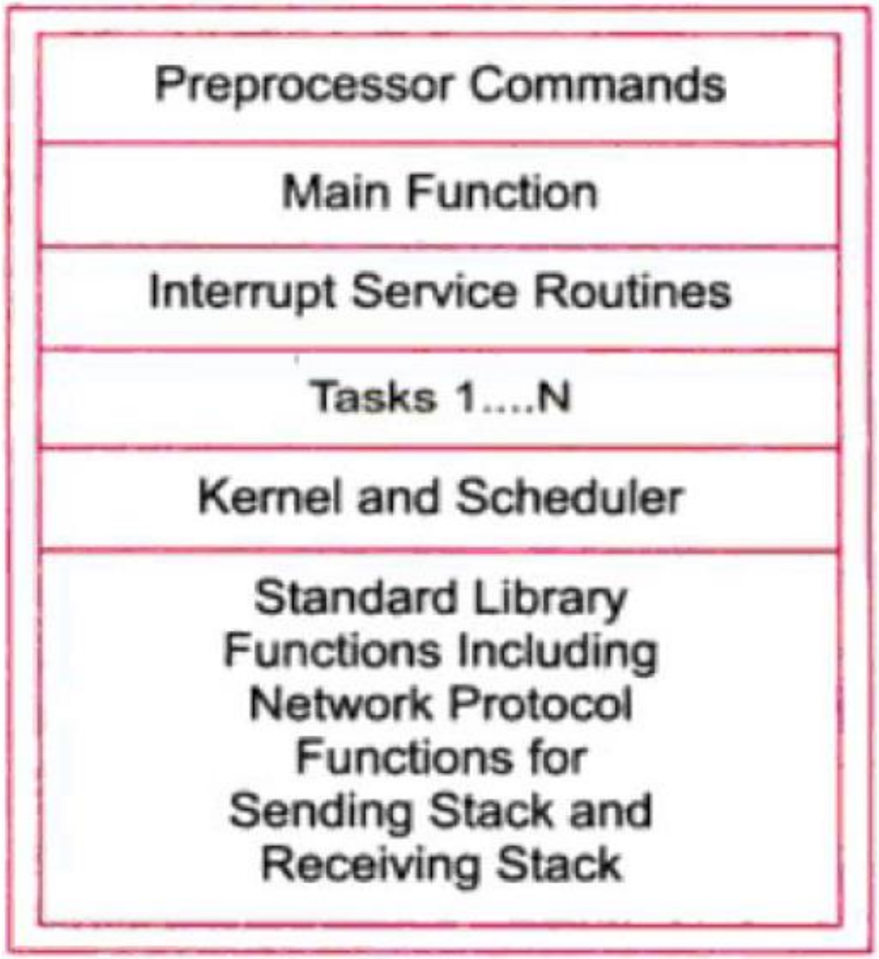
Embedded Hardware Units and Devices.....

4. Software in High Level Language:

- Coding in assembly is very time consuming, so software is developed in high level languages like C or C++ or Visual C++ or Java.

Figure shows different programming layers in a typical embedded C software.

- Processor Commands,
- Main Function,
- Interrupt Service Routines,
- Multiple Tasks (task functions),
- Kernel and Scheduler,
- Standard Library Functions, Protocol handling and Stack functions.

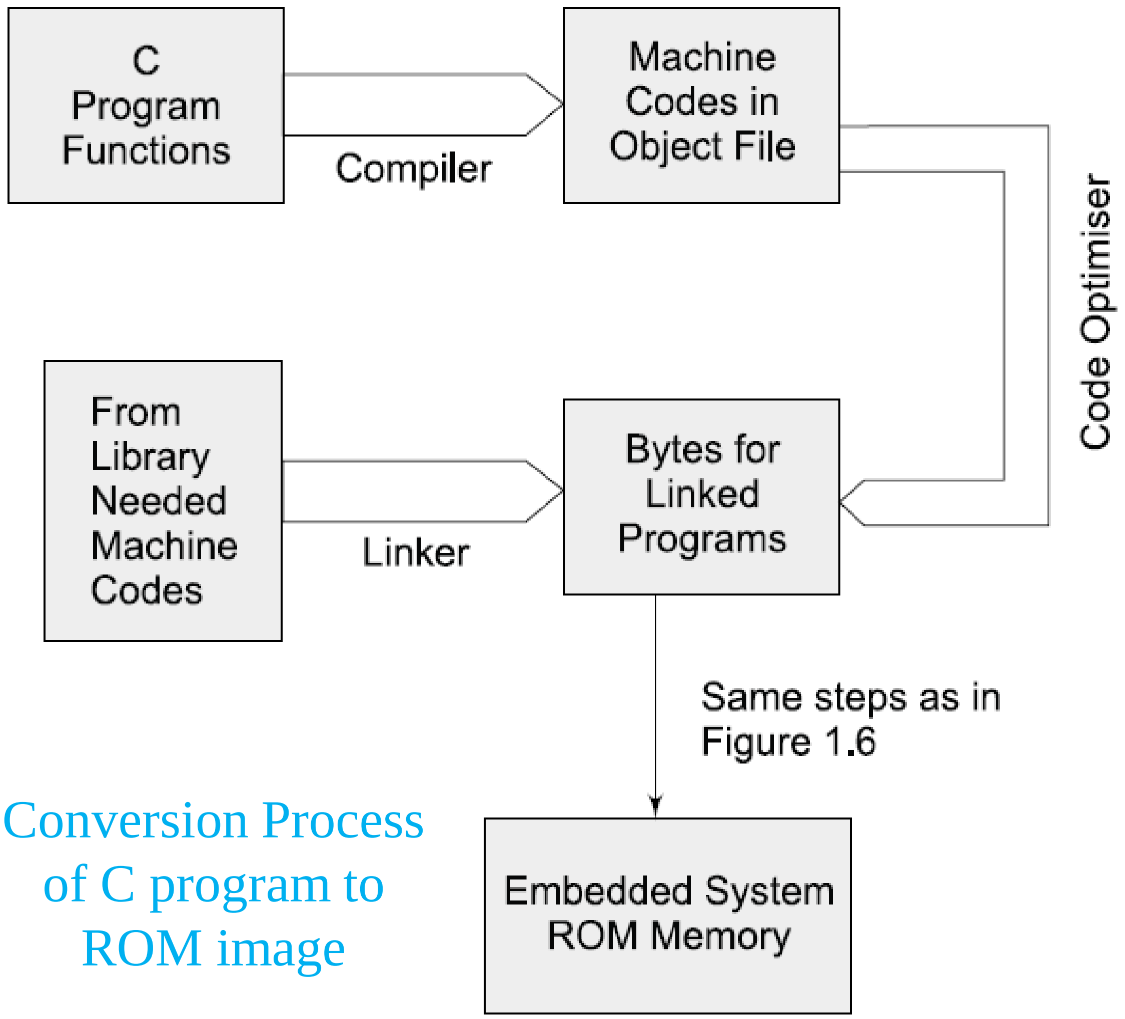

Preprocessor Commands
Main Function
Interrupt Service Routines
Tasks 1....N
Kernel and Scheduler
Standard Library Functions Including Network Protocol Functions for Sending Stack and Receiving Stack

Embedded Hardware Units and Devices…..

4. Software in High Level Language…

The steps followed when using high level language before finally burned at the ROM

- ‘Compiler’ – Generates the object codes. It assembles the code according to the processor instruction set and other specifications.
- ‘Code Optimizer’ – Optimizes the codes before linking.
- ‘Linker’ – Link the codes with the other codes. The linked file in binary for run is known as exe file.
- ‘Loader’ – Relocating the codes after finding the physical memory addresses available.
- The file for ROM image is created for the targeted hardware.


C Program Functions
Compiler
Machine Codes in Object File
Code Optimiser
From Library Needed Machine Codes
Linker
Bytes for Linked Programs
Same steps as in Figure 1.6
Embedded System ROM Memory

Conversion Process of C program to ROM image

Embedded Hardware Units and Devices…..

5. Program Models for Software Designing:

- The program design task is simplified if a program is modeled.
- The different models that are employed during the design process of embedded software are as follows
  - Sequential Programming Model
  - Object Oriented Programming Model
  - Control and Data flow graphs or Synchronous Data Flow (SDF) Graph or Multi Thread Graph (MTG) Model
  - Finite State Machine for data path
  - Multithreaded Model for Concurrent Processing of processes or thread or tasks

- UML (Universal Modelling Language) is a modelling language for Object oriented programming.

Embedded Hardware Units and Devices.....

6. Software for Concurrent processing and Scheduling of Multiple Tasks and ISRs using an RTOS:

- An embedded system program is designed using multiple processes or multitasks or multithreads.
- These are processed by the OS not sequentially but concurrently.
- Concurrent processing tasks can be interrupted for running the ISRs and higher priority task preempts the running of lower priority tasks.
- OS provides for Processes, memory, device, ports, network, file system management.
- OS software have scheduling functions for all the processes (tasks, ISRs and device drivers).
- RTOS is required when the task and ISRs may have real-time constraints and deadlines for finishing the tasks.
- RTOS provides inter process communication functions and these are highly complex.
- **Popular RTOSs:** OS µCOS-II, VxWorks, Windows CE, RTLinux,...

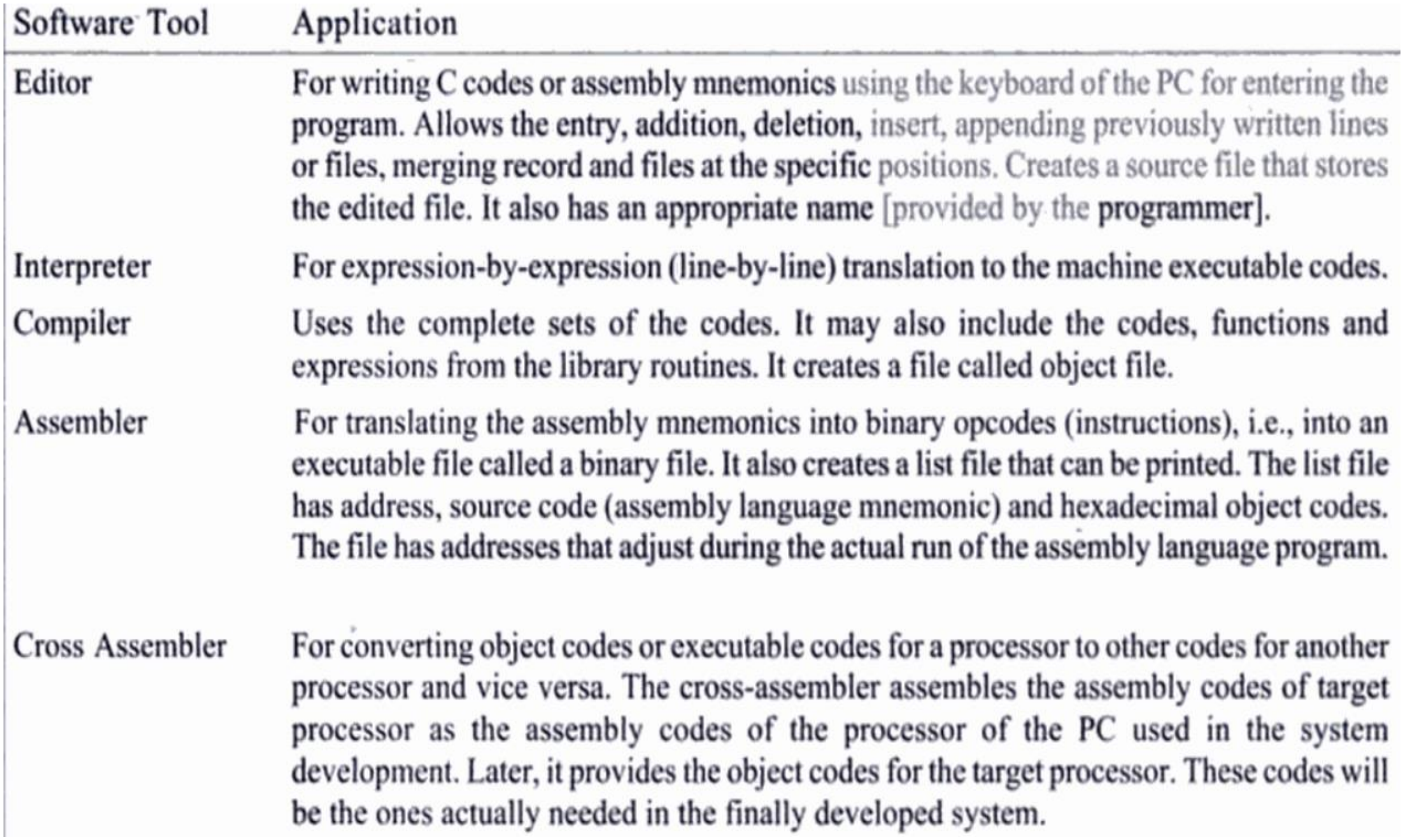
Embedded Hardware Units and Devices.....

7. Software for Device Drivers and Device Management in an Operating System:

- An embedded system is designed to perform multiple functions and has to control multiple **physical** and **virtual** devices.
- Physical devices – keypad, LCD display or touch screen, memory stick (flash memory), wireless networking device, parallel port and network card….
- Virtual devices – file, pipe, RAM disk, socket, …
- Device consisting 3 components – i) Control register or word, ii) Status register or word and iii) Device mechanism that controls the device actions.
- A **device driver** is software for opening, connecting, controlling (configuring), reading, writing and closing actions of the device.
- A Driver controls 3 functions – i) initializing CWR, ii) calling ISR or set flags and iii) Resetting the status flag after service.

Embedded Hardware Units and Devices…..

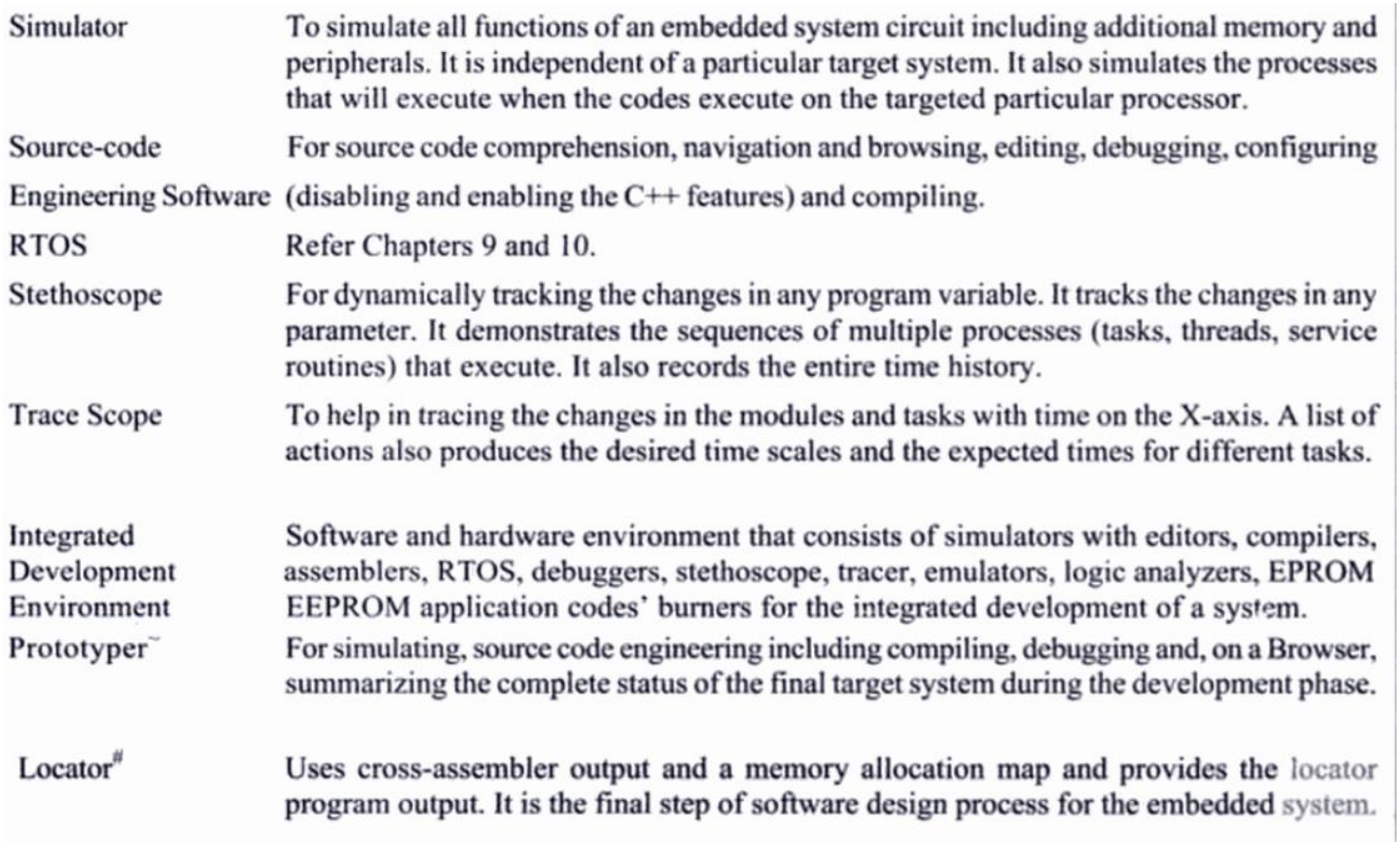
8. Software Tools for Designing an ES:



Software Tool	Application
Editor	For writing C codes or assembly mnemonics using the keyboard of the PC for entering the program. Allows the entry, addition, deletion, insert, appending previously written lines or files, merging record and files at the specific positions. Creates a source file that stores the edited file. It also has an appropriate name [provided by the programmer].
Interpreter	For expression-by-expression (line-by-line) translation to the machine executable codes.
Compiler	Uses the complete sets of the codes. It may also include the codes, functions and expressions from the library routines. It creates a file called object file.
Assembler	For translating the assembly mnemonics into binary opcodes (instructions), i.e., into an executable file called a binary file. It also creates a list file that can be printed. The list file has address, source code (assembly language mnemonic) and hexadecimal object codes. The file has addresses that adjust during the actual run of the assembly language program.
Cross Assembler	For converting object codes or executable codes for a processor to other codes for another processor and vice versa. The cross-assembler assembles the assembly codes of target processor as the assembly codes of the processor of the PC used in the system development. Later, it provides the object codes for the target processor. These codes will be the ones actually needed in the finally developed system.

Embedded Hardware Units and Devices…..

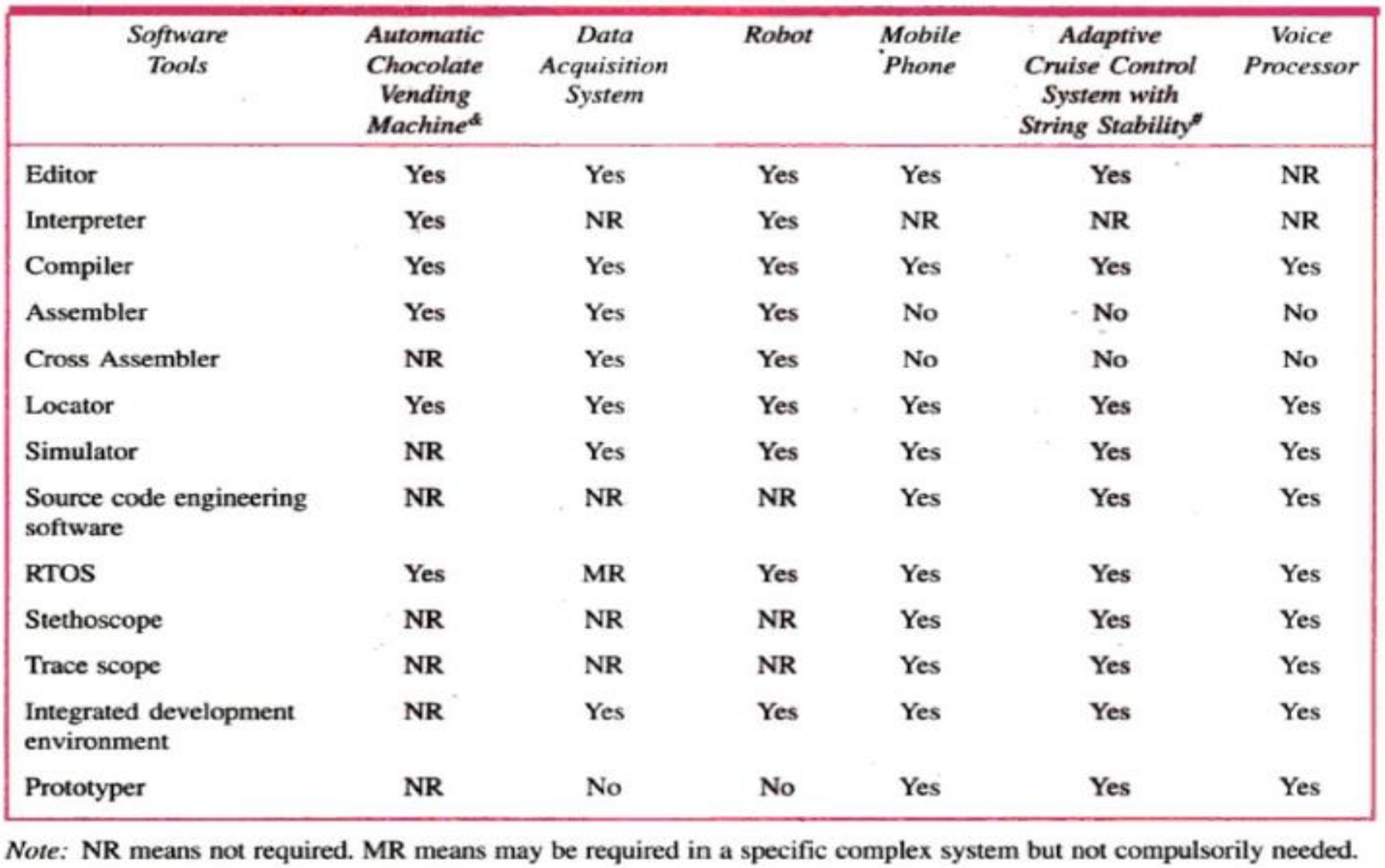
8. Software Tools for Designing an ES……



Simulator	To simulate all functions of an embedded system circuit including additional memory and peripherals. It is independent of a particular target system. It also simulates the processes that will execute when the codes execute on the targeted particular processor.
Source-code Engineering Software	For source code comprehension, navigation and browsing, editing, debugging, configuring (disabling and enabling the C++ features) and compiling.
RTOS	Refer Chapters 9 and 10.
Stethoscope	For dynamically tracking the changes in any program variable. It tracks the changes in any parameter. It demonstrates the sequences of multiple processes (tasks, threads, service routines) that execute. It also records the entire time history.
Trace Scope	To help in tracing the changes in the modules and tasks with time on the X-axis. A list of actions also produces the desired time scales and the expected times for different tasks.
Integrated Development Environment	Software and hardware environment that consists of simulators with editors, compilers, assemblers, RTOS, debuggers, stethoscope, tracer, emulators, logic analyzers, EPROM EEPROM application codes' burners for the integrated development of a system.
Prototyper~	For simulating, source code engineering including compiling, debugging and, on a Browser, summarizing the complete status of the final target system during the development phase.
Locator#	Uses cross-assembler output and a memory allocation map and provides the locator program output. It is the final step of software design process for the embedded system.

Embedded Hardware Units and Devices.....

9. Software Tools Required in Exemplary Cases:



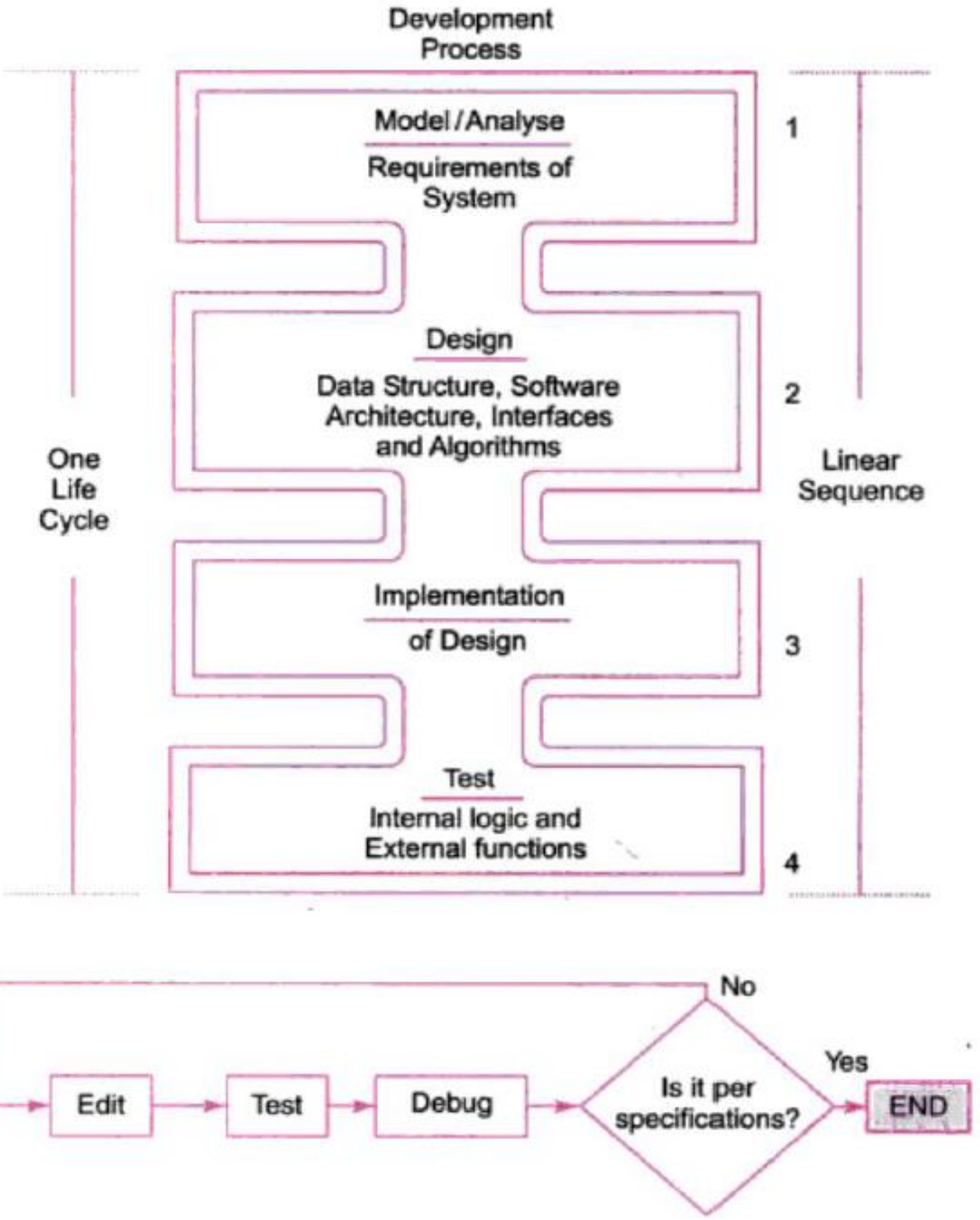
Software Tools	*Automatic Chocolate Vending Machine*[&]	*Data Acquisition System*	*Robot*	*Mobile Phone*	*Adaptive Cruise Control System with String Stability*[#]	*Voice Processor*
Editor	Yes	Yes	Yes	Yes	Yes	NR
Interpreter	Yes	NR	Yes	NR	NR	NR
Compiler	Yes	Yes	Yes	Yes	Yes	Yes
Assembler	Yes	Yes	Yes	No	No	No
Cross Assembler	NR	Yes	Yes	No	No	No
Locator	Yes	Yes	Yes	Yes	Yes	Yes
Simulator	NR	Yes	Yes	Yes	Yes	Yes
Source code engineering software	NR	NR	NR	Yes	Yes	Yes
RTOS	Yes	MR	Yes	Yes	Yes	Yes
Stethoscope	NR	NR	NR	Yes	Yes	Yes
Trace scope	NR	NR	NR	Yes	Yes	Yes
Integrated development environment	NR	Yes	Yes	Yes	Yes	Yes
Prototyper	NR	No	No	Yes	Yes	Yes

Note: NR means not required. MR means may be required in a specific complex system but not compulsorily needed.

Design Process in ES:

Concepts used in design process:

1. Abstraction,
2. Hw/Sw architecture,
3. Extra functional properties,
4. System related family of design,
5. Modular design,
6. Mapping,
7. User Interface design,
8. Refinements



Design Metrics in ES:

There are several design metrics for an embedded system.

1. Power Dissipation,
2. Performance,
3. Process deadlines,
4. User interfaces,
5. Size,
6. Cost,
7. Flexibility,
8. Prototype,
9. Development time,
10. Time to Market,
11. System and user safety,
12. Maintenance.

Design Metrics in ES…….

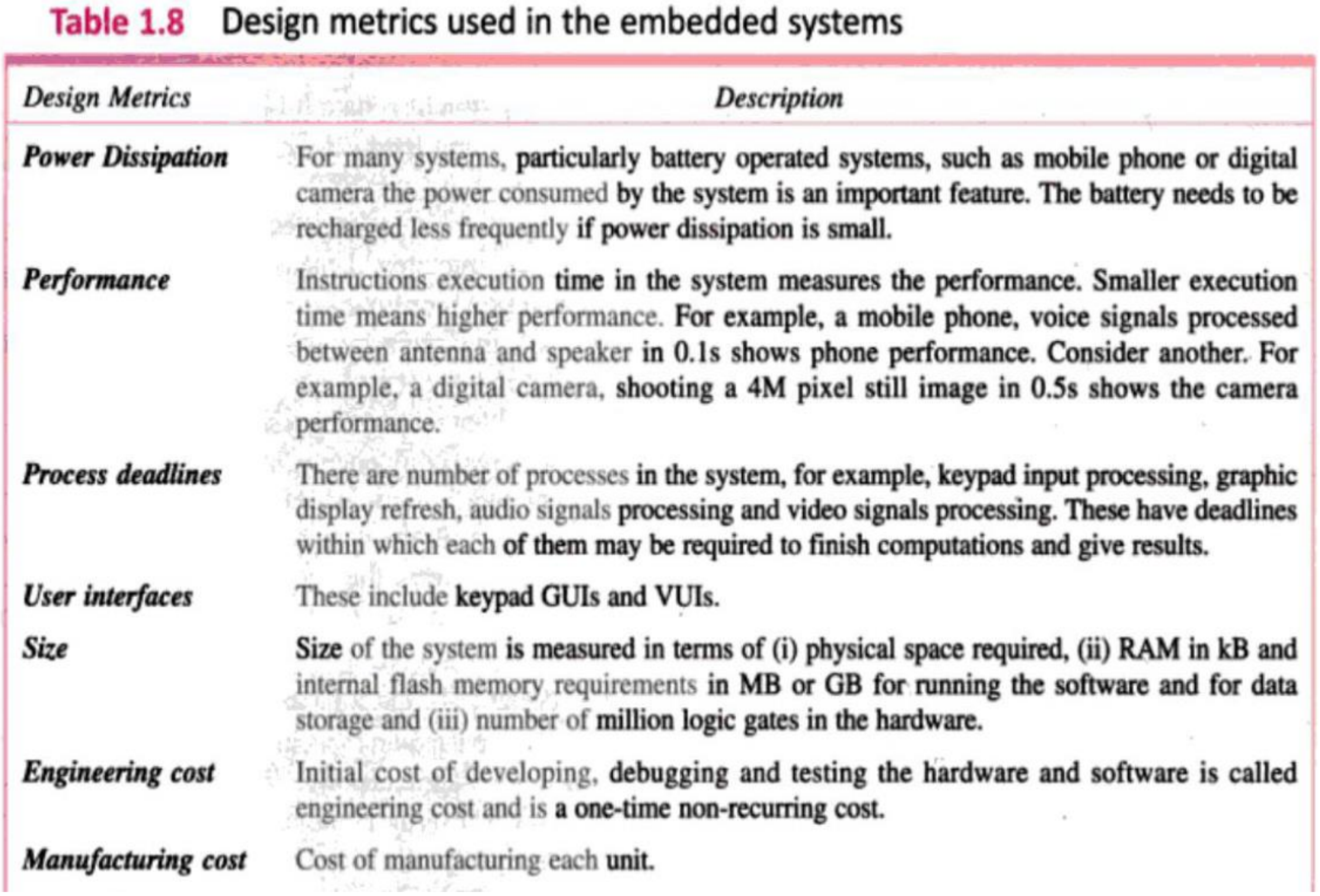
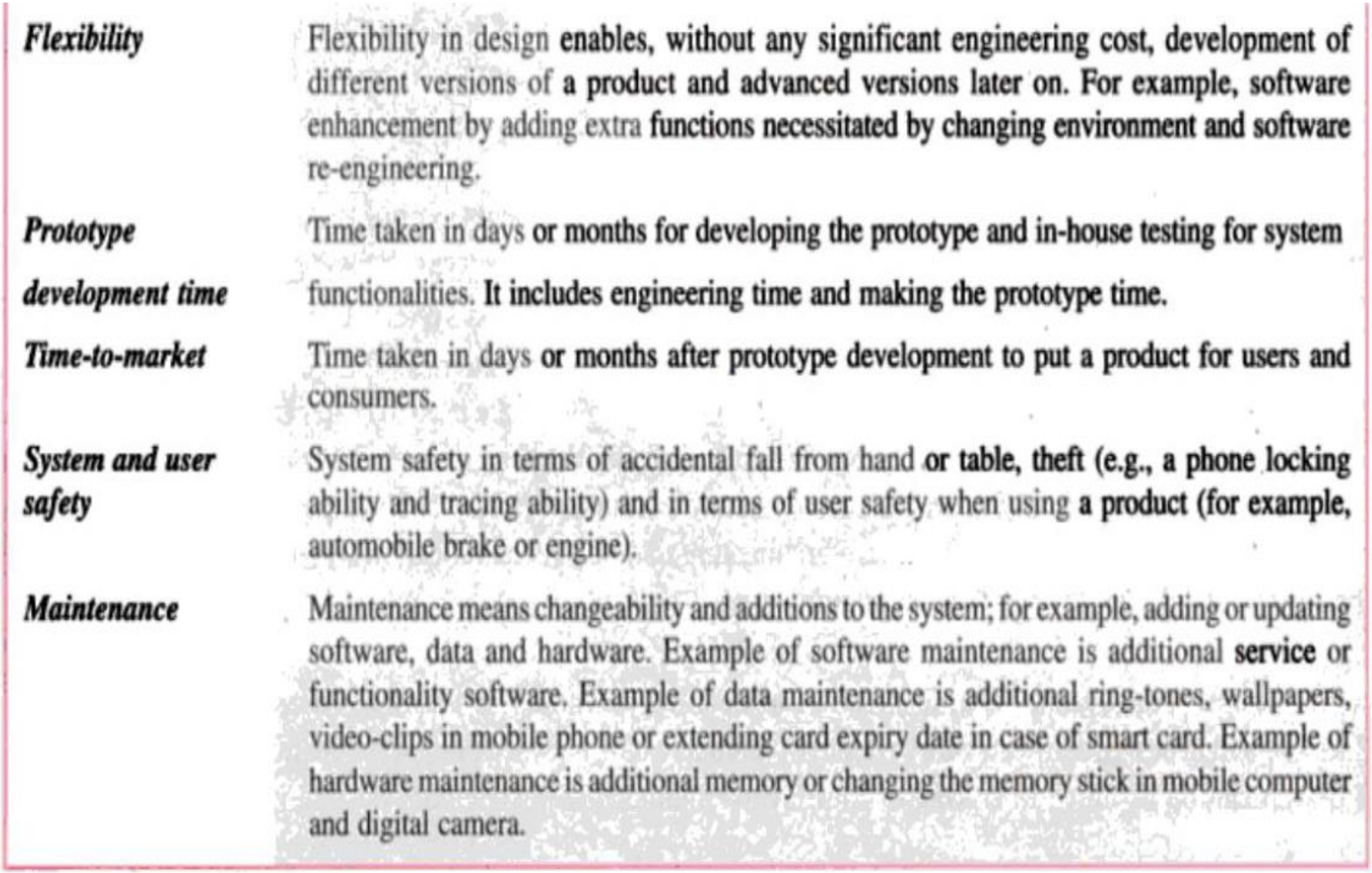


Table 1.8 Design metrics used in the embedded systems

Design Metrics	*Description*
Power Dissipation	For many systems, particularly battery operated systems, such as mobile phone or digital camera the power consumed by the system is an important feature. The battery needs to be recharged less frequently if power dissipation is small.
Performance	Instructions execution time in the system measures the performance. Smaller execution time means higher performance. For example, a mobile phone, voice signals processed between antenna and speaker in 0.1s shows phone performance. Consider another. For example, a digital camera, shooting a 4M pixel still image in 0.5s shows the camera performance.
Process deadlines	There are number of processes in the system, for example, keypad input processing, graphic display refresh, audio signals processing and video signals processing. These have deadlines within which each of them may be required to finish computations and give results.
User interfaces	These include keypad GUIs and VUIs.
Size	Size of the system is measured in terms of (i) physical space required, (ii) RAM in kB and internal flash memory requirements in MB or GB for running the software and for data storage and (iii) number of million logic gates in the hardware.
Engineering cost	Initial cost of developing, debugging and testing the hardware and software is called engineering cost and is a one-time non-recurring cost.
Manufacturing cost	Cost of manufacturing each unit.

Design Metrics in ES…….



Flexibility	Flexibility in design enables, without any significant engineering cost, development of different versions of a product and advanced versions later on. For example, software enhancement by adding extra functions necessitated by changing environment and software re-engineering.
Prototype development time	Time taken in days or months for developing the prototype and in-house testing for system functionalities. It includes engineering time and making the prototype time.
Time-to-market	Time taken in days or months after prototype development to put a product for users and consumers.
System and user safety	System safety in terms of accidental fall from hand or table, theft (e.g., a phone locking ability and tracing ability) and in terms of user safety when using a product (for example, automobile brake or engine).
Maintenance	Maintenance means changeability and additions to the system; for example, adding or updating software, data and hardware. Example of software maintenance is additional service or functionality software. Example of data maintenance is additional ring-tones, wallpapers, video-clips in mobile phone or extending card expiry date in case of smart card. Example of hardware maintenance is additional memory or changing the memory stick in mobile computer and digital camera.

Challenges in ES design:

Challenges during the embedded system design process are

- Amount and type of hardware needed,
- Optimizing Power Dissipation and Consumption,
- Process deadlines,
- Flexibility and upgradability,
- Reliability.

Challenges in ES design……

Amount and type of hardware needed:

- Optimizing the requirement of microprocessors, ASIPs and single purpose processors in the system.
- Optimizing according to the performance, power dissipation, cost and other design metrics are the challenges in a system design.
- A designer also chooses appropriate hardware based on the design metrics
  - Memory - RAM, ROM or internal and external flash or secondary memory,
  - Peripherals and devices,
  - Internal and external ports and busses,
  - Power source or battery

Challenges in ES design……

Optimizing Power Dissipation and Consumption:

Power consumption during the operational and idle state of system should be optimal.

Following methods are used to meet the design challenges.

- **Clock rate reduction**
  - Power dissipation reduces 0.25μW per 100kHz of reduced clock rate. (Reduce 8000kHz to 100kHz reduces 200μW power)
  - If clock rate is reduced, then the computation will take longer time.
  - By reducing the clock rate the advantages are
    - ✓ Power loss due to heat generation reduces,
    - ✓ RF interference also reduces.
- **Voltage reduction**
  - Portable or hand held devices, compared to 5V operation a CMOS circuit power dissipation reduces by one sixth (~2V/5V)2 in 2.0V operation.
  - Thus the time needed for recharging the battery increase by a factor of 6.
- **Wait, stop and cache disable instruction**
  - By using of 'Wait' and 'Stop' instructions and disabling or controlling certain units when not needed.
  - Caches-when not necessary and keep in disconnected state those structure units that are not needed during a particular software-portion execution, for example, display screen, timers or IO units.

Challenges in ES design……

Process deadlines

- Meeting the deadline of all processes in the system while keeping the memory, power dissipation, processor clock rate and cost at minimum is a challenge.

Flexibility and upgradability

- Ability in design while keeping the cost minimum and without any significant engineering cost is a challenge.
- Flexibility and upgradability allow different versions or advanced versions of a product to be introduced in the market later on.

Challenges in ES design……

Reliability

- Designing reliable product by appropriate design and thorough testing, verification and validation is a challenge.
  - **Testing** – to find errors and to validate that the implemented software is as per the specifications and requirements to get reliable product.
  - **Verification** – refers to an activity to ensure that specific functions are correctly implemented.
  - **Validation** – refers to an activity to ensure that the system that has been created is as per requirements agreed upon at the analysis phase, and to ensure its quality.

Host and Target Machines:

Host:

- A computer system on which all the programming tools run.
- Where the embedded software is developed, compiled, tested, debugged, optimized, and prior to its translation into target device.
- Host is generally a PC or laptop or workstation.

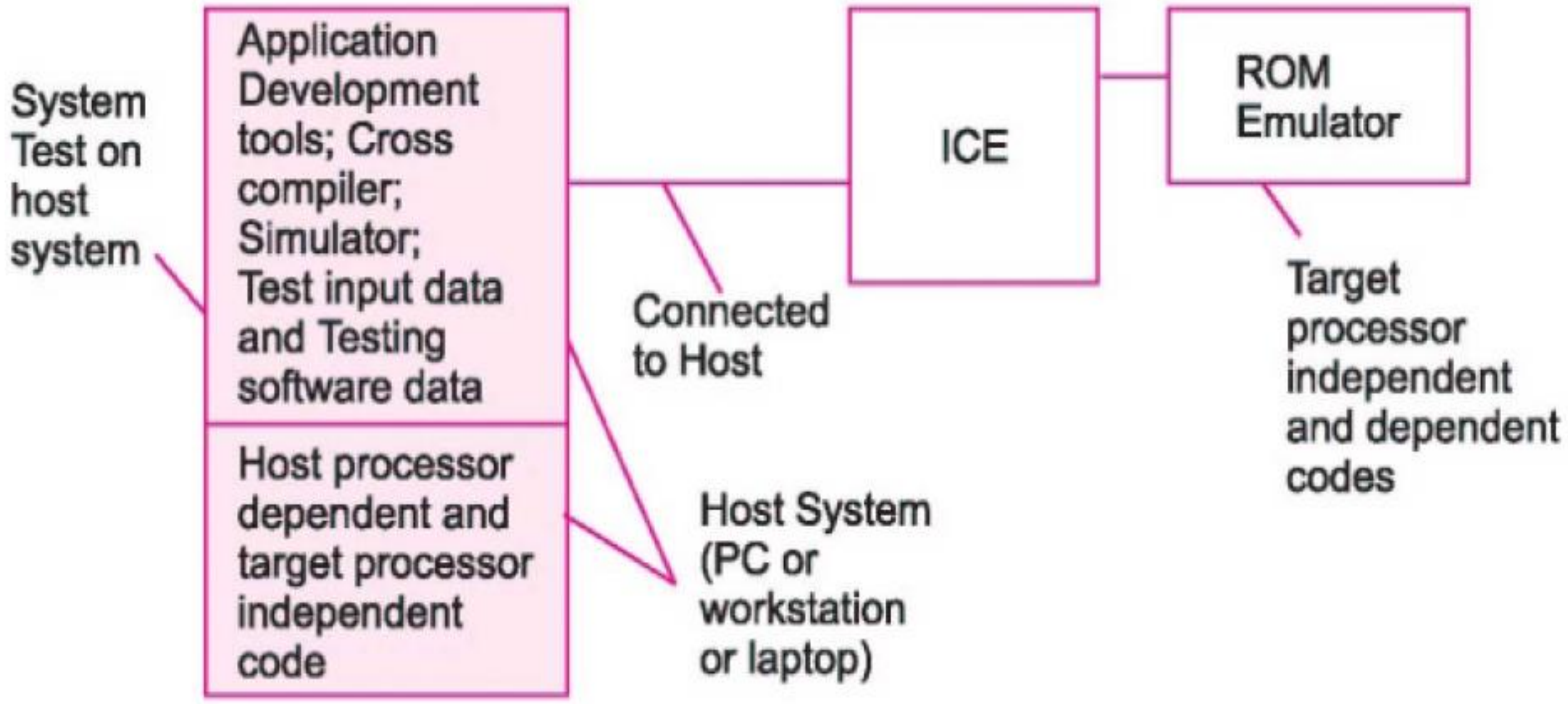
Target:

- After writing the program, compiled, assembled and linked, then moved to target.
- After development, code is cross-compiled, cross-assembled, linked into target processor instruction set and located into the target.
- Target is a actual hardware for the embedded system under development.

Host and Target Machines……

- **Code** has two parts: hardware independent code and hardware dependent code.
- Port and devices - have fixed addresses on hardware.

Host and Target Test Systems During Development Process


System Test on host system
Application Development tools; Cross compiler; Simulator; Test input data and Testing software data
Host processor dependent and target processor independent code
Connected to Host
Host System (PC or workstation or laptop)
ICE
ROM Emulator
Target processor independent and dependent codes

Host and Target Machines……

Host-Target System Development Approach:

- During development process, a host system is used.
- Then locating and burning the codes in the target board.
- Target board hardware and software later copied to get the final embedded system.
- Final system function exactly as the one tested and debugged and finalized during the development process.

Host system at PC or Workstation or Laptop:

- High performance processor with caches,
- Large RAM memory,
- ROM BIOS (read only memory basic input-output system),
- Very large memory on disk,
- Keyboard, display monitor and mouse,
- Network connection,
- Program development kit for a high level language program or IDE,
- Host processor compiler and cross compiler,
- Cross assembler.

Host and Target Machines……

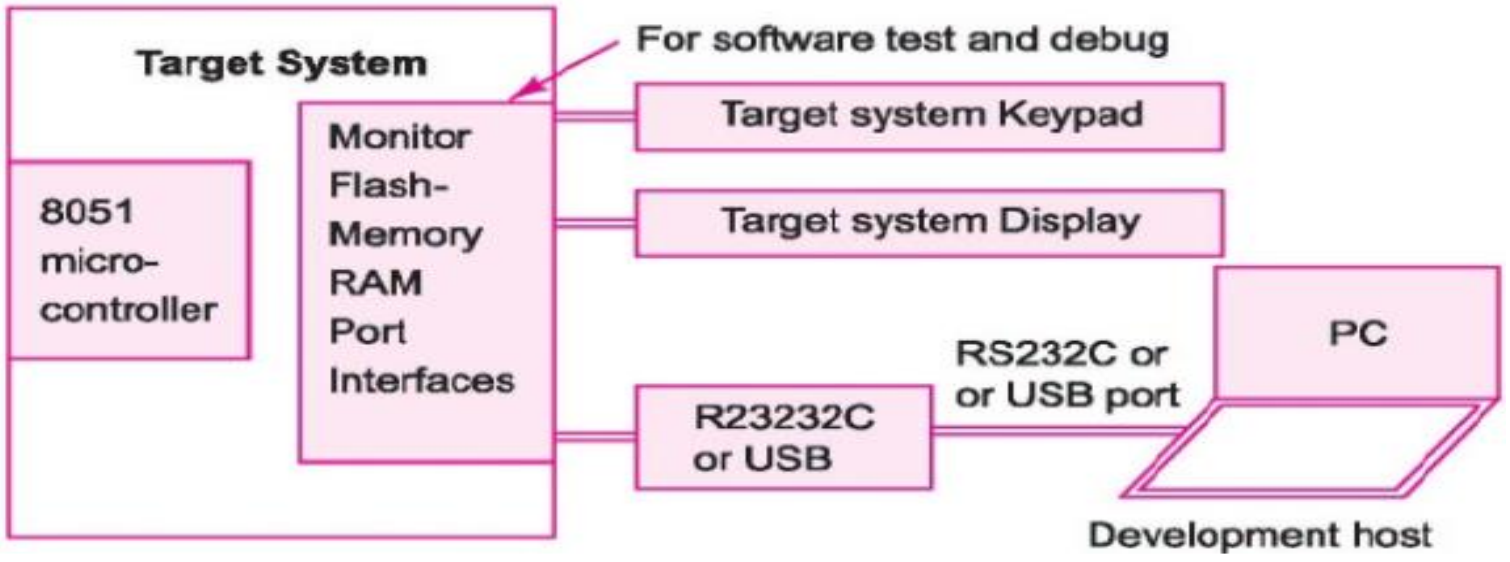
Target and Final systems:

- Target system differs from a final system,
- Target system interfaces with the computer as well works as a standalone system,
- In target system might be repeated downloading of the codes during the development phase,
- Target system copy made that later on functions as embedded system,
- Designer later on simply copies it into final system or product,
- Final system may employs ROM in place of flash, EEPROM or EPROM in embedded system,

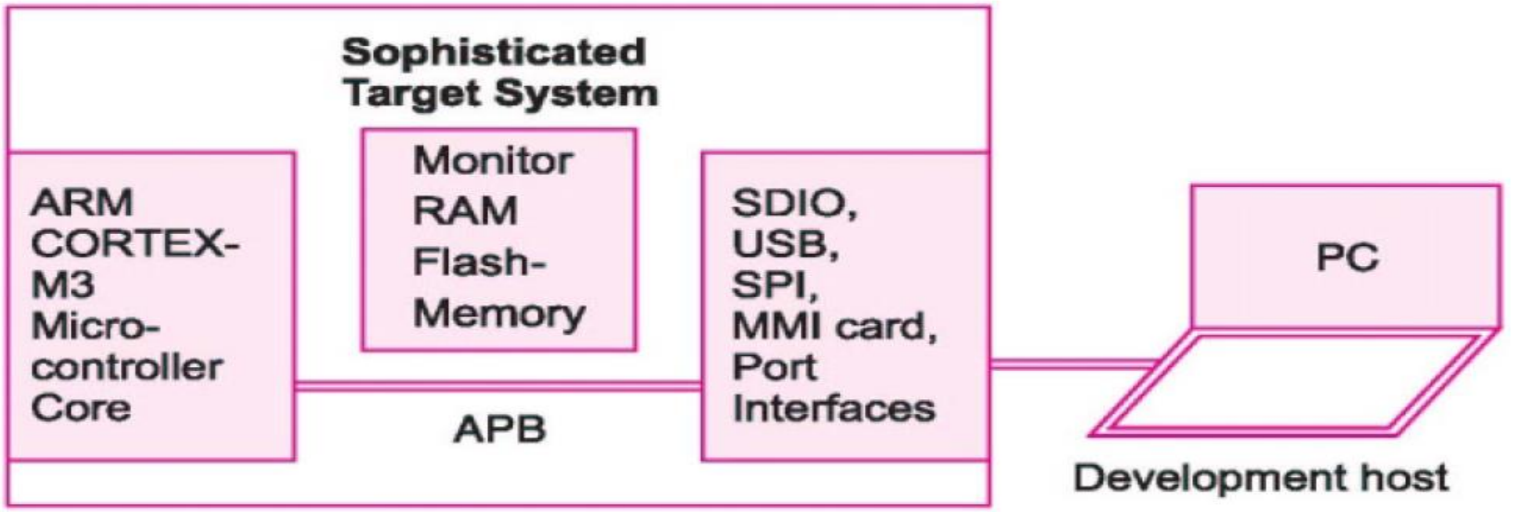
- Examples: Phillips LPC 21xx, ARM Powered STR710 System development boards.

Host and Target Machines……

Target System Board



Sophisticated Target System



Host and Target Machines…….

Host System	Target Computer System
Writing, editing a program, compiling it, linking it, debugging it are done on host system	After the completion of programming work, it is moved from host system to target system.
It is also referred as Work Station	No other name
Software development is done in host system for embedded system	Developed software is shifted to customer from host
Compiler, linker, assembler, debugger are used	Cross compiler is also used
Unit testing on host system ensures software is working properly	By using cross compiler, unit testing allows to recompile code ,execute, test on target system
Stubs are used	Real libraries
Programming centric	Customer centric

Host and Target Machines……

Testing Steps at Host Machine:

- Initial Tests - each module or segment at initial stage itself and on host itself.
- Test data - all possible combinations of data designed and taken as test data.
- Exception condition tests - all possible exceptions for the test.
- Tests-1: hardware independent code.
- Tests-2: scaffold software, software running on host the target dependent codes and which have same start of code and port and device addresses as at the hardware.
    - Instructions - given from file or keyboard inputs.
    - Outputs - at host's LCD display and saves at file.
- Test Interrupt Service routines hardware independent part - sections of interrupt service routines are called, which are hardware independent and tested.
- Test Interrupt Service routines hardware dependent part.
- Timer tests - Hardware dependent code timing functions, clock tick set, counts get, counts put, delay.
- Assert-Macro tests - insert the codes in the program that check whether a condition or a parameter actually turns true or false.
    - If it turns false - the program stops.
    - Use the assert macro at different critical places in the application program

Getting ES in to Target System:

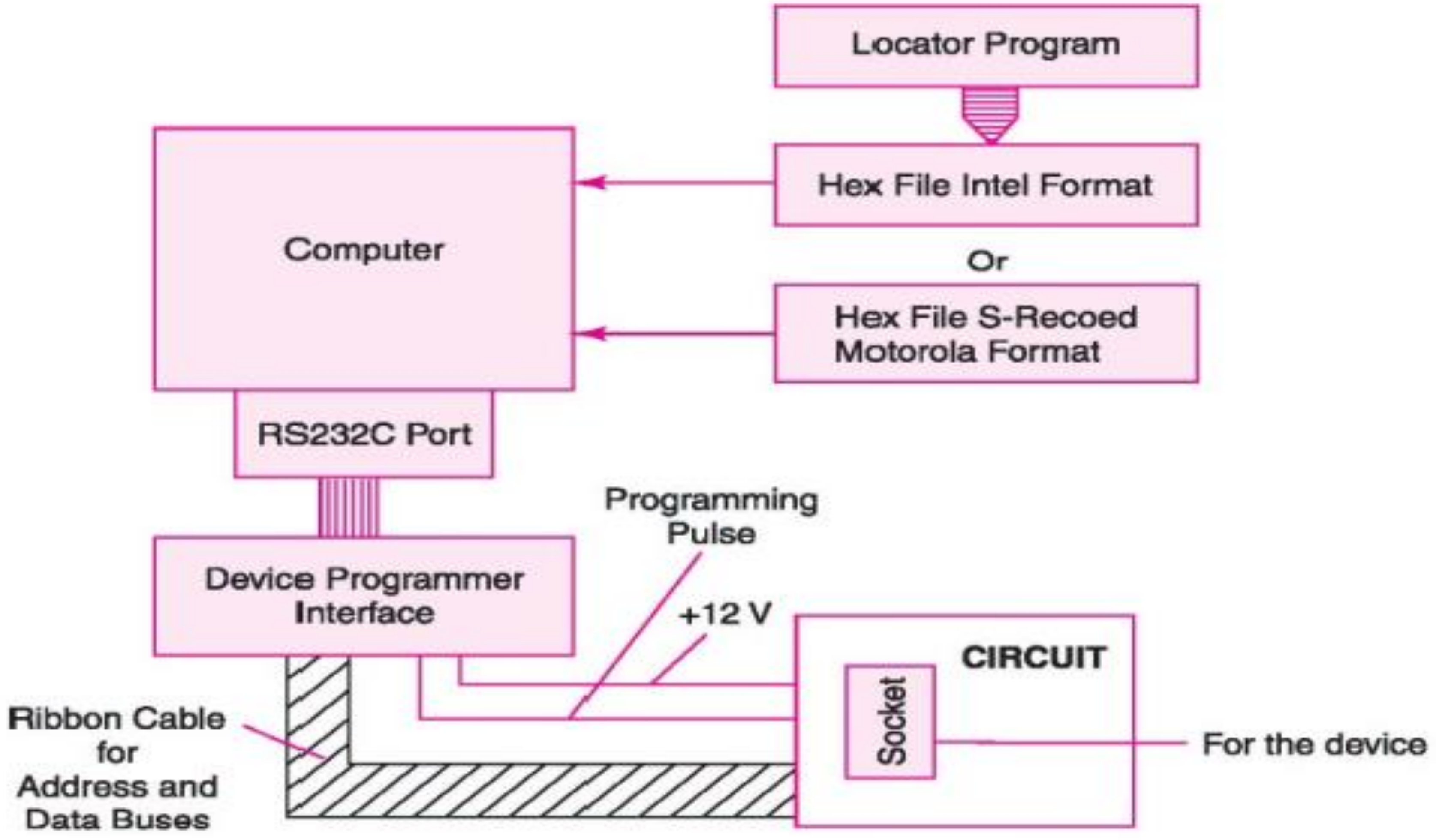
Getting Embedded Software into the Target System using Device Programmer.

Device PROM or Flash Programmer:

- Also called laboratory programmer,
- A programming system for a device,
- Device selectable,
- Device may be a PROM or EPROM chip or a flash or a unit in a microcontroller or PLA, GAL or PLC,
- Selected device inserts into a socket,
- Programmed (burned the codes) by transfer of the bytes for each address using the software at the host.

Getting ES in to Target System……

Burning in of the application software codes, data and tables using a device programmer.



Getting ES in to Target System……

Software of Device Programmer:

- Runs at a host system,
- The system interconnects with the socket and device programmer circuit usually through a serial port (UART or USB),
- Running at the host uses an input file containing the from the locator software output records,
- File reflects the final design and has a boot program plus the compressed, record, which the processor decompresses before the embedded system processor starts execution,
- Bootstrap program is the program to start up a system.
- An IDE incorporates the device programmer within it.

Use of Device Programmer for Downloading the Finalized Codes into PROM or flash: Burning

- A process that places the codes.
- Codes downloaded, according to ROM image (locator output)
- Burning done in the laboratory using a device programmer into an erased
- EPROM or EEPROM or PROM or flash

Getting ES in to Target System……

Programming Steps of the Device Programmer:

i. Applies the A0 to A19 bits as needed at a selected address input of the array of cells.

ii. Applies as inputs, the D0 to D7 bits that are meant for that address.

iii. Applies a high voltage to make programming feasible for the needed duration in microseconds.

iv. Applies a programming pulse for a sufficient duration to cause fusing of the desired links in the array, to convert a '1' to '0'.

v. Switches off high voltage

vi. Applies a next higher address than the previous one.

vii. Repeats the above steps (ii) to (iv) for writing (converting) the logic states of the D0 to D7 bits at the current instance at the new address, and

viii. Continues till a cell array at the last desired address programmed.

REFERENCE BOOKS

- Raj Kamal, "Embedded Systems", 2nd Edition, Pearson, 2009.
- K.V.K.K. Prasad, "Embedded Real-Time Systems: Concepts, Design & Programming", Dream Tech Press, 2005.
- Marilyn Wolf, "Computers as Components - Principles of Embedded Computing System Design", Third Edition, Morgan Kaufmann Publisher-Elsevier, 2012.
- Raj Kamal, "Microcontroller, Programming and Design", 2nd Edition, Pearson, 2012.
- Andrew N Sloss, Dominic Symes and Chris Wright, "ARM system developers guide", Morgan Kaufmann Publisher-Elsevier, 2008.
- Michael J Pont, "Embedded C", Pearson Education, 2007.
- Steve Furber, "ARM System on Chip Architecture", Addison Wesley, 2nd ed., 2000.
- Wang K.C., Embedded and Real-Time Operating Systems, Springer, 2017.
- Frank Vahid and Tony Givargis, Embedded System Design: A Unified Hardware/Software Introduction, John Wiley & Sons, Student edition, 2006.

Thank You