Outline

- **Chapter 1: Introduction to Embedded Systems**
  - What is an Embedded System?
  - Embedded Systems v/s General Computing Systems
  - Classification of Embedded Systems
  - Major application areas of Embedded Systems
  - Purpose of Embedded Systems
- **Chapter 2: The Typical Embedded System**
  - Core of the Embedded system
  - Memory

Chapter 1: Introduction to Embedded Systems

What is an Embedded system?

- An embedded system is an electronic/electro-mechanical system designed to perform a specific function and is a combination of both hardware and firmware (software).

Embedded systems vs. General Computing systems

General Purpose Computing System	Embedded System
A system which is a combination of a generic hardware and a General Purpose OS for executing a variety of applications	A system which is a combination of special purpose hardware and embedded OS for executing a specific set of applications
Contains a General Purpose OS (GPOS)	May or may not contain an operating system for functioning
Applications are alterable (programmable) by the user	The firmware of the embedded system is pre-programmed and is non-alterable by the end user
Performance is the key deciding factor in the selection of the system. Always, 'faster is better'	Application specific requirements(like performance, power requirements, memory usage, etc.) are the key deciding factors
Less/not at all tailored towards reduced operating power requirements	Highly tailored to take advantage of the power saving modes supported by the hardware and the operating system
Response requirements are not time-critical	For certain category of embedded systems like mission critical systems, the response time requirement is highly critical
Need not be deterministic in execution behaviour	Execution behaviour is deterministic for certain types of embedded systems like 'Hard Real Time' systems

Classification of Embedded systems

- It is possible to have multitude of classifications for embedded systems, based on different criteria. Some of the criteria used in the classification of embedded systems are as follows:
    1. Based on generation
    2. Based on Complexity and performance requirements
    3. Based on deterministic behaviour
    4. Based on triggering

- The classification based on deterministic system behaviour is applicable for 'Real Time' systems.
- The application/task execution behaviour for an embedded system can be either ***deterministic*** or ***non-deterministic***.
- Based on the execution behaviour, Real Time embedded systems are classified into ***Hard*** and ***Soft***.
- Embedded Systems which are 'Reactive' in nature (Like process control systems in industrial control applications) can be classified ***based on the trigger***.
    - Reactive systems can be either ***event triggered*** or ***time triggered***.

Classification Based on Generation

- This classification is based on the order in which the embedded processing systems evolved from the first version to where they are today.

➢ **First Generation:**

- The early embedded systems were built around 8-bit microprocessors like 8085 and Z80, and 4-bit microcontrollers.
- Simple in hardware circuits with firmware developed in assembly code.
- **Examples:** Digital telephone keypads, stepper motor control units etc.

- **Second Generation:**
  - These are embedded systems built around 16-bit microprocessors and 8-bit or 16-bit microcontrollers, following the first generation embedded systems.
  - The instruction set for the processor/controllers in this generation were much more complex and powerful than the first generation processor/controllers.
  - Some of the second generation embedded systems contained embedded operating systems for their operation.
  - **Examples:** Data Acquisition Systems, SCADA systems, etc.

- **Third Generation:**
  - With advances in processor technology, embedded system developers started making use of powerful 32-bit processors and 16-bit microcontrollers for their design.
  - A new concept of application and domain specific processors/controllers like Digital Signal Processors (DSP) and Application Specific Integrated Circuits (ASICs) came into picture.
  - The instruction set of processors became more complex and powerful and the concept of instruction pipelining also evolved.
  - The processor market was flooded with different vendors. Processors like Intel Pentium, Motorola 68K, etc. gained attention in high performance embedded requirements.
  - Dedicated embedded real time and general purpose operating systems entered into the embedded market.
  - Embedded systems spread its ground to areas like robotics, media, industrial process control, networking etc.

- **Fourth Generation:**
  - The advent of System on Chips (SoC), reconfigurable processors and multicore processors are bringing high performance, tight

integration and miniaturisation into the embedded device market.

- The SoC technique implements a total system on a chip by integrating different functionalities with a processor core on an integrated circuit.
- The fourth generation embedded systems are making use of high performance real time embedded operating systems for their functioning.
- Examples: Smart phone devices, mobile internet devices (MIDs) etc.

Classification Based on Complexity and Performance

- This classification is based on the complexity and system performance requirements. According to this classification, embedded systems can be grouped into the following:
  - **Small-Scale Embedded Systems:**
    - Embedded systems which are simple in application needs and where the performance requirements are not time critical fall under this category.
    - These embedded systems are usually built around low performance and low cost 8 or 16-bit microprocessors/microcontrollers.
    - These embedded systems may or may not contain an operating system for its functioning.
    - **Example:** An electronic toy is a typical example of a small-scale embedded system.
  - **Medium-Scale Embedded Systems:**
    - Embedded systems which are slightly complex in hardware and firmware (software) requirements fall under this category.
    - These embedded systems are usually built around medium performance, low cost 16 or 32-bit microprocessors/microcontrollers or digital signal processors.

- They usually contain an embedded operating system (either general purpose or real time operating system) for functioning.

➢ **Large-Scale Embedded Systems / Complex Embedded Systems:**

- Embedded systems which involve highly complex hardware and firmware requirements fall under this category.
- They are employed in mission critical applications demanding high performance.
- Such systems are commonly built around high performance 32 or 64-bit RISC processors/controllers or Reconfigurable System on Chip (RSoC) or multicore processors and programmable logic devices.
- They may contain multiple processors/controllers and co-units/hardware accelerators for offloading the processing requirements from the main processor of the system.
- Decoding/encoding of media, cryptographic function implementation, etc, are examples for processing requirements which can be implemented using a co-processor/hardware accelerator.
- These systems usually contain a high performance Real Time Operating System (RTOS) for task scheduling, prioritisation, and management.

Major application areas of Embedded Systems

The application areas and the products in the embedded domain are countless. A few of the important domains and products are listed below:

1. ***Consumer electronics:*** Camcorders, cameras, etc.
2. ***Household appliances:*** Television, DVD players, washing machine, fridge, microwave oven, etc.
3. ***Home automation and security systems:*** Air conditioners, sprinklers, intruder detection alarms, fire alarms etc.

4. ***Automotive industry:*** Anti-lock breaking systems (ABS), engine control, ignition systems, automatic navigation systems, etc.
5. ***Telecom:*** Cellular telephones, telephone switches, handset multimedia applications, etc.
6. ***Computer peripherals:*** Printers, scanners, fax machines, etc.
7. ***Computer networking systems:*** Network routers, switches, hubs, firewalls, etc.
8. ***Health Care:*** Different kinds of scanners, EEG,ECG machines etc**.**
9. ***Measurement & Instrumentation:*** Digital multimeters, digital CROs, etc.
10. ***Banking & Retail:*** Automatic Teller Machines (ATM) and currency counters, etc.
11. ***Card Readers:*** Barcode, smart card readers, hand held devices etc.
12. ***Wearable Devices:*** Health and Fitness Trackers, Smartphone Screen extension for notifications etc.
13. ***Cloud Computing and Internet of Things (IOT).***

Purpose of Embedded Systems

- Each embedded system is designed to serve the purpose of any one or a combination of the following tasks:
    1. **Data collection/Storage/Representation**
    2. **Data communication**
    3. **Data (signal) processing**
    4. **Monitoring**
    5. **Control**
    6. **Application specific user interface**

1. Data collection/Storage/Representation:

- Embedded systems designed for the purpose of data collection performs acquisition of data from the external world.
- Data collection is usually done for storage, analysis, manipulation, and transmission.

- The term "data" refers to all kinds of information, i.e., text, voice, image, video, electrical signals and any other measurable quantities.
- Data can be either analog or digital.
    - Embedded systems with analog data capturing techniques collect data directly in the form of analog signals.
    - Embedded systems with digital data collection mechanism converts the analog signal to corresponding digital signal using ADCs and then collects the binary equivalent of the analog data.
        - If data is digital, it can be directly captured without any additional interface by digital embedded systems.
- The collected data may be stored directly in the system or may be transmitted to some other systems or it may be processed by the system or it may be deleted instantly after giving a meaningful representation. These actions are *purely dependent on the purpose for which the embedded system is designed.*
- Embedded systems designed for ***pure measurement applications without storage***, used in control and instrumentation domain, collects data and gives a meaningful representation of the collected data by means of graphical representation or quantity value and deletes the collected data when new data arrives at the data collection terminal.
    - **Examples:** Analog and digital CROs without storage memory, any measuring equipment used in the medical domain for monitoring without storage functionality etc.
- Some embedded systems store the collected data for processing and analysis.
    - Such systems incorporate a built-in/plug-in storage memory for storing the captured data.
- Some of the embedded systems give the user a meaningful (graphical/quantitative) or audible means using display units (LCD, LED, etc), buzzers, alarms, etc.

- Examples: Measuring instruments with storage memory and monitoring instruments with storage memory used in medical applications.

- A ***digital camera*** is atypical example of an embedded system with data collection/storage/representation of data.
  - Images are captured and the captured image may be stored within the memory of the camera. The captured image can also be presented to the user through a graphic LCD unit.

2. Data Communication:

- Embedded data communication systems are deployed in applications ranging from complex satellite communication systems to simple home networking systems.
- The data collected by an embedded terminal may require transferring of the same to some other system located remotely.
  - The transmission is achieved either by a wire-line or by a wireless medium.
    - A wireless medium offers cheaper connectivity solutions and make the communication link free from the hassle of wire bundles.
- Data can either be transmitted by analog means or by digital means.
  - Modern industry trends are settling towards digital communication.
- The data collecting embedded terminal itself can incorporate data communication units like wireless modules (Bluetooth, Zigbee, Wi-Fi, etc) or wire-line modules (RS-232C, USB, TCP/IP, etc.).
- Certain embedded systems act as a dedicated transmission unit between the sending and receiving terminals, offering sophisticated functionalities like data packetizing, encrypting and decrypting.
  - Network hubs, routers, switches, etc are typical examples of dedicated data transmission embedded systems.

- They act as mediators in data communication and provide various features like data security, monitoring etc.

3. Data (Signal) Processing:

- The data (voice, image, video, electrical signals, and other measurable quantities) collected by embedded systems may be used for various kinds of data processing.
- Embedded systems with signal processing functionalities are employed in applications demanding signal processing like speech coding, synthesis, transmission applications, etc.
  - A digital hearing aid is a typical example of an embedded system employing data processing.
    - Digital hearing aid improves the hearing capacity of hearing impaired persons.

4. Monitoring:

- Embedded systems falling under this category are specifically designed for monitoring purpose.
- Almost all embedded products coming under the medical domain are with monitoring functions only.
  - They are used for determining the state of some variables using input sensors. They cannot impose control over variables.
- A very good example is the electrocardiogram(ECG) machine for monitoring the heartbeat of a patient.
  - The machine is intended to do the monitoring of the heartbeat. It cannot impose control over the heartbeat.
- Some other examples of embedded systems with monitoring function are measuring instruments like digital CRO, digital

multimeters, logic analysers, etc. used in control & Instrumentation applications.

- They are used for knowing (monitoring) the status of some variables like current, voltage, etc.

5. Control:

- Embedded systems with control functionalities impose control over some variables according to the change in input variables.
- A system with control functionality contains both sensors and actuators.
- Sensors are connected to the input port for capturing the changes in environmental variables or measuring variable.
- The actuators connected to the output port are controlled according to the changes in input variable to put an impact on the controlling variable to bring the controlled variable to the specified range.
- Air conditioner system used in our home to control the room temperature to a specified limit is a typical example for embedded system for control purpose.
  - An air conditioner contains a room temperature-sensing element (sensor) which may be a thermistor and a handheld unit for setting up the desired temperature.
  - The handheld unit may be connected to the central embedded unit residing inside the air conditioner through a wireless link or through a wired link.
  - The air compressor unit acts as the actuator. The compressor is controlled according to the current room temperature and the desired temperature set by the end user.
  - Here, input variable is the current room temperature and the controlled variable is also the room temperature. The controlling variable is cool air flow by the compressor unit. If the controlled variable and the input variable are not at the same value, the

controlling variable tries to equalise them through taking actions on the cool air flow.

6. Application Specific User Interface:

- These are embedded systems with application-specific user interfaces like buttons, switches, keypad, lights, bells, display units, etc.
  - Mobile phone is an example for this. In mobile phone, the user interface is provided through the keypad, graphic LCD module, system speaker, vibration alert, etc.

Chapter 2: The Typical Embedded System

Core of the Embedded System

- Embedded systems are domain and application specific and are built around a central core.
- The core of the embedded system falls into any one of the following categories:
  1. General Purpose and Domain Specific Processors
     - Microprocessors
     - Microcontrollers
     - Digital Signal Processors
  2. Application Specific Integrated Circuits (ASICs)
  3. Programmable Logic Devices (PLDs)
  4. Commercial off-the-shelf Components (COTS)

1. General Purpose and Domain Specific Processors

- Almost 80% of the embedded systems are processor/controller based.
- The processor may be a microprocessor or a microcontroller or a digital signal processor, depending on the domain and application.
- Most of the embedded systems in the industrial control and monitoring applications make use of the commonly available microprocessors or microcontrollers.
- Domains which require signal processing such as speech coding, speech recognition, etc. make use of special kind of digital signal processors supplied by manufacturers like, Analog Devices, Texas Instruments, etc.

1.1 Microprocessor

- A Microprocessor is a silicon chip representing a central processing unit (CPU), which is capable of performing arithmetic as well as

logical operations according to a pre-defined set of instructions, which is specific to the manufacturer.

- In general, the CPU contains the Arithmetic and Logic Unit (ALU), control unit and working registers.
- A microprocessor is a dependent unit and it requires the combination of other hardware like memory, timer unit and interrupt controller, etc. for proper functioning.
- The first microprocessor, **Intel 4004** is a 4-bit processor which was released in 1971.
- The Features of Intel 4004 are:
  - It has 1K data memory
  - It has 12-bit PC
  - It has 4K program memory
  - It consists of 16 4-bit general purpose registers
  - It supports 46 instructions
  - It ran at a clock speed of 740 kHz.
  - It was designed for olden day's calculator
- In1972, 14 more instructions were added to 4004 instruction set and the program space is upgraded to 8K. Also, interrupt capabilities were added to it and it is renamed as **Intel 4040**.
- **Intel 8008** replaced 4040. It was similar to 4040 but with 14-bit PC and it served as a terminal controller.
- In 1974, Intel launched the first 8 bit processor, the **Intel 8080**, with 16-bit address bus and PC, and 7 8-bit registers.
  - It was the most commonly used processors for industrial control and other embedded applications in the 1975s.
  - Systems made out of it were bulky and were lacking compactness.
- Motorola released **Motorola 6800** with a different architecture and instruction set compared to Intel 8080.
- In 1976, Intel came up with the upgraded version of 8080-**Intel 8085**, with two newly added instructions, three interrupt pins and serial

I/O. Clock generator and bus controller circuits were built-in and power supply part was modified to a single +5 V supply.

- In 1976, Zilog came up with **Z80 processor**. It was an improved version of Intel's 8080 processor, maintaining the original 8080 architecture and instruction set with an 8-bit data bus and a 16-bit address bus and was capable of executing all instructions 0f 8080.
  - It included 80 more new instructions and it brought out the concept of register banking by doubling the register set.
  - It also included two sets of index registers for flexible design.
- Twentieth century witnessed a fast growth in processor technology.
  - 16, 32 and 64-bit processors came into the place of conventional 8-bit processors.
  - Today processors with clock speeds over 3.0 GHz are available in the market.
- Intel, AMD, Freescale, TI, Cyrix, NVIDIA, Qualcomm, MediaTek, etc. are the key players in the processor market. Intel still leads the market with cutting edge technologies in the processor industry.
- Different instruction set and system architecture are available for the design of a microprocessor.
  - Harvard and Von-Neumann are the two common system architectures for processor design.
  - RISC and CISC are the two common Instruction Set Architectures (ISA) available for processor design.

1.2 General Purpose Processor (GPP) vs. Application-Specific Instruction Set Processor (ASIP)

- GPP is a processor designed for general computational tasks.
  - The Processor running inside our laptop or desktop (Pentium4 etc.) is a typical example.
- Due to the high volume production, the per unit cost for a chip is low compared to ASIC or other specific ICs.
- A typical GPP contains an ALU, Control Unit.

- ASIPs are processors with architecture and instruction set optimized to specific-domain/application requirements like network processing, automotive, telecom, media applications, digital signal processing, control applications, etc.
- ASIPs fill the architectural spectrum between general purpose processors and Application Specific Integrated Circuits (ASICs).
- Most of the embedded systems are built around ASIPs.
  - Some microcontrollers (Automotive AVR etc) , digital signal processors, etc. are examples for ASIPs.
- ASIPs incorporate a processor and on-chip peripherals, demanded by the application requirement, program and data memory.

1.3 Microcontrollers

- A microcontroller is a highly integrated chip that contains a CPU, RAM, special and general purpose register arrays, on chip ROM/Flash memory for program storage, timer and interrupt control units and dedicated I/O ports.
- Microcontrollers can be considered as a super set of microprocessors.
- Since a microcontroller contains all the necessary functional blocks for independent working, they found greater place in the embedded domain in place or microprocessor.
- Microcontrollers are cheap, cost effective and are readily available in the market.
- Texas Instrument's TMS 1000 released in 1974 is considered as the world's first microcontroller.
- TI followed Intel's 4004/4040, 4-bit processor design and added some amount of RAM, program storage memory (ROM) and I/O support on a single chip, there by eliminated the requirement of multiple hardware chips for self-functioning.
  - Provision to add custom instructions to the CPU was another innovative feature.

- In 1977, Intel entered the microcontroller market with a family of controllers coming under one umbrella named **MCS-48 family**. Intel 8048 was the most prominent member in the MCS-48 family.
- In 1980s Intel came up with most popular and powerful 8-bit microcontroller, 8051 which was put under the family MCS-51.
- Almost 75% of the microcontrollers used in the embedded domain were **8051 family** based controllers during the 1980-90s.
    - Due to the low cost, wide availability, memory efficient instruction set, mature development tools and Boolean processing (bit manipulation operation) capability, 8051 family derivative microcontrollers are much used in high-volume consumer electronic devices, entertainment industry and other gadgets where cost-cutting is essential.
- Another important family of microcontrollers used in industrial control and embedded applications is the **PIC family** microcontrollers from Microchip Technologies.
    - It is a high performance RISC microcontroller complementing the CISC features of 8051.
- Some embedded system applications require only 8-bit controller whereas some embedded applications requiring superior performance and computational needs demand **16/32-bit microcontrollers.**
    - Infineon, Freescale, Philips, Atmel, Maxim, Microchip etc. are the key suppliers of 16-bit microcontrollers.
- Philips tried to extend the 8051 family microcontrollers to use for 16-bit applications by developing the Philips XA (eXtended Architecture) microcontroller series.
- 8-bit microcontrollers are commonly used in embedded systems where the processing power is not a big constraint.
- High processing speed microcontroller families like ARM Cortex M series are also available in the market, which provides solution to applications requiring hardware acceleration and high processing capability.

- Freescale, Zilog, Cypress, Infineon, EPSON, Texas Instruments, Toshiba, NXP, Microchip, Analog Devices, INTEL, Maim, Sharp, Silicon Laboratories, Atmel, Renesas, ST Micro Electronics, LAPIS, HOLTEK, CYROD, etc are the key players in the microcontroller market.
- Out of these manufacturers, **Atmel** has got special significance.
  - They are the manufacturers of a variety of Flash memory based microcontrollers.
    - The Flash memory technique helps in fast reprogramming of the chip and thereby reduces the product development time.
- Atmel also provide another special family of microcontroller called **AVR**, an 8-bit RISC Flash microcontroller, fast enough to execute powerful instructions in a single clock cycle and provide the latitude you need to optimize power consumption.
- The instruction set architecture of a microcontroller can be either RISC or CISC.
- Microcontrollers are designed for either general purpose application requirement (general purpose controller) or domain-specific application requirement (application specific instruction set processor).
  - The **Intel 8051 microcontroller** is a typical example for a general purpose microcontroller.
  - The **automotive AVR microcontroller family** from Atmel Corporation is a typical example for ASIP specifically designed for the automotive domain.

1.4 Microprocessor vs. Microcontrollers

- The following table summarizes the differences between a microcontroller and microprocessor.

Microprocessor	Microcontroller
A silicon chip representing a central processing unit (CPU), which is capable of performing arithmetic as well as logical operations according to a predefined set of instructions	A highly integrated chip that contains a CPU, scratchpad RAM, special and general purpose register arrays, on chip ROM/FLASH memory for program storage, timer and interrupt control units and dedicated I/O ports.
It is a dependent unit. It requires the combination of other chips like timers, program and data memory chips, interrupt controllers, etc. for functioning	It is a self-contained unit and it doesn't require external interrupt controller, timer, UART, etc. for its functioning
Most of the time general purpose in design and operation	Mostly application-oriented or domain-specific
Doesn't contain a built-in I/O port. The I/O port functionality needs to be implemented with the help of external programmable peripheral interface chips like 8255	Most of the processors contain multiple built-in I/O ports which can be operated as a single 8 or 16 or 32-bit port or as individual port pins
Limited power saving options compared to microcontrollers	Includes lot of power saving features

1.5 Digital Signal Processors

- DSPs are powerful special purpose 8/16/32 bit microprocessors designed specifically to meet the computational demands and power constraints of today's embedded audio, video, and communications applications.
- DSPs are 2 to 3 times faster than the general purpose microprocessors in signal processing applications. This is because of the architectural difference between the two.
- DSPs implement algorithms in hardware which speeds up the execution whereas general purpose processors implement the

algorithm in firmware and the speed of execution depends primarily on the clock for the processors.

- In general, DSP can be viewed as a microchip designed for performing high speed computational operations for 'addition', 'subtraction', 'multiplication' and 'division'.

A typical DSP incorporates the following key units:

- **Program Memory:** Memory for storing the program required by DSP to process the data.
- **Data Memory:** Working memory for storing temporary variables and data/signal to be processed.
- **Computational Engine:** performs the signal processing in accordance with the stored program memory. Computational Engine incorporates many specialized arithmetic units and each of them operates simultaneously to increase the execution speed. It also incorporates multiple hardware shifters for shifting operands and thereby saves execution time.
- **I/O unit:** Acts as an interface between the outside world and DSP. It is responsible for capturing signals to be processed and delivering the processed signals.

- Audio video signal processing, telecommunication and multimedia applications are typical examples where DSP is employed.
- Digital signal processing employs a large amount of real-time calculations.
- Sum of products (SOP) calculation, convolution, fast fourier transform (FFT), discrete fourier transform (DFT), etc, are some of the operations performed by digital signal processors.
- Blackfin processors from Analog Devices is an example of DSP which delivers breakthrough signal processing performance and power efficiency while also offering a full 32-bit RISC MCU programming model.

1.6 RISC vs. CISC Processors/Controllers

RISC (Reduced Instruction Set Computer)	CISC (Complex Instruction Set Computer)
Less number of instructions	Greater number of instructions
Instruction pipelining and increased execution speed	Generally no instruction pipelining feature
Single, fixed length instructions	Variable length instructions
Operations are performed on registers only, the only memory operations are load and store	Operations are performed on registers or memory depending on the instruction
A large number of registers are available	Limited number of general purpose registers
Orthogonal instruction set (allows each instruction to operate on any register and use any addressing mode)	Non-orthogonal instruction set (all instructions are not allowed to operate on any register and use any addressing mode. It instruction-specific)
Less silicon usage and pin count	More silicon usage since more additional decoder logic is required to implement the complex instruction decoding
With Harvard Architecture	Can be Harvard or Von-Neumann Architecture
Hardwired control unit organization	Microprogrammed control unit organization
Atmel AVR microcontroller is an example for a RISC processor and its instruction set contains only 32 instructions	The original version of 8051 micorcontroller (e.g. AT89C51) is a CISC controller and its instruction set contains 255 instructions

1.7 Harvard vs. Von-Neumann Processor/Controller Architecture

- The term Harvard and Von-Neumann refers to the processor architecture design.

Von-Neumann/Princeton Architecture:

- Microprocessors/controllers based on the Von-Neumann architecture shares a single common bus for fetching both instructions and data.
- Program instructions and data are stored in a common main memory.
- Von-Neumann architecture based processors/controllers first fetch an instruction and then fetch the data to support the instruction from code memory. The two separate fetches slows down the controller's operation.
- Von-Neumann architecture is also referred as Princeton architecture, since it was developed by the Princeton University.

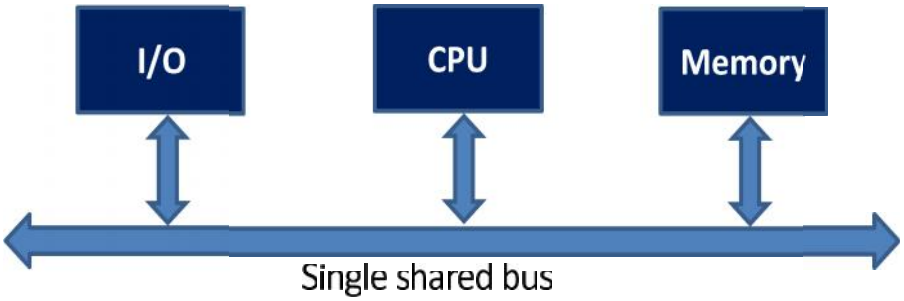

Figure: Von-Neumann Architecture

Harvard Architecture:

- Microprocessors/controllers based on the Harvard architecture will have separate data bus and instruction bus. This allows the data transfer and program fetching to occur simultaneously on both buses.
- With Harvard architecture, the data memory can be read and written while the program memory is being accessed. These separated data memory and code memory buses allow one instruction to execute while the next instruction is fetched.

Figure: Harvard Architecture

Differences between Harvard and Von-Neumann Architecture

Harvard Architecture	Von-Neumann Architecture
Separate buses for instruction and data fetching	Single shared bus for instruction and data fetching
Easier to pipeline, so high performance can be achieved	Low performance compared to Harvard architecture
Comparatively high cost	Cheaper
Since data memory and program memory are stored physically in different locations, no chances for accidental corruption of program memory	Since data memory and program memory are stored physically in the same chip, chances for accidental corruption of program memory.

1.8 Big-Endian vs. Little-Endian Processors/Controllers

- Endianness specifies the order in which the data is stored in the memory by processor operations in a multibyte system.

- **Little-endian** means the lower-order byte of the data is stored in memory at the lowest address, and the higher-order byte at the highest address.
- For example, a 4 byte integer **Byte3 Byte2 Byte1 Byte0** will be stored in the memory as shown below:

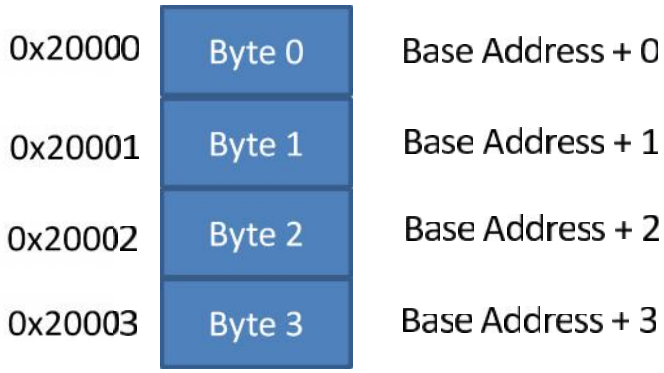

Figure: Little-Endian Operation

- **Big-endian** means the higher-order byte of the data is stored in memory at the lowest address, and the lower-order byte at the highest address.
- For example, a 4 byte integer **Byte3 Byte2 Byte1 Byte0** will be stored in the memory as shown below:

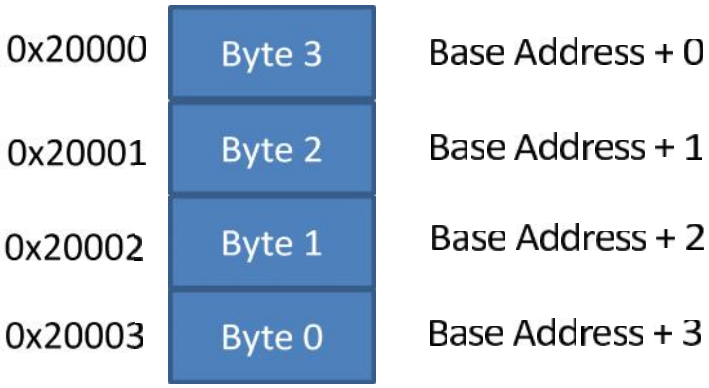

Figure: Big-Endian Operation

1.9 Load Store Operation and Instruction Pipelining

- The RISC processor instruction set is orthogonal, meaning it operates on registers.

- The memory access related operations are performed by the special instructions ***load*** and ***store***.
  - If the operand is specified as memory location, the content of it is loaded to a register using the ***load instruction***.
  - The ***store instruction*** stores data from a specified register to a specified memory location.

The concept of Load Store Architecture is illustrated with the following example:

- Suppose x, y, z are memory locations and we want to add the contents of x and y and store the result in location z. Under the load store architecture the same is achieved with 4 instructions as shown in below figure.

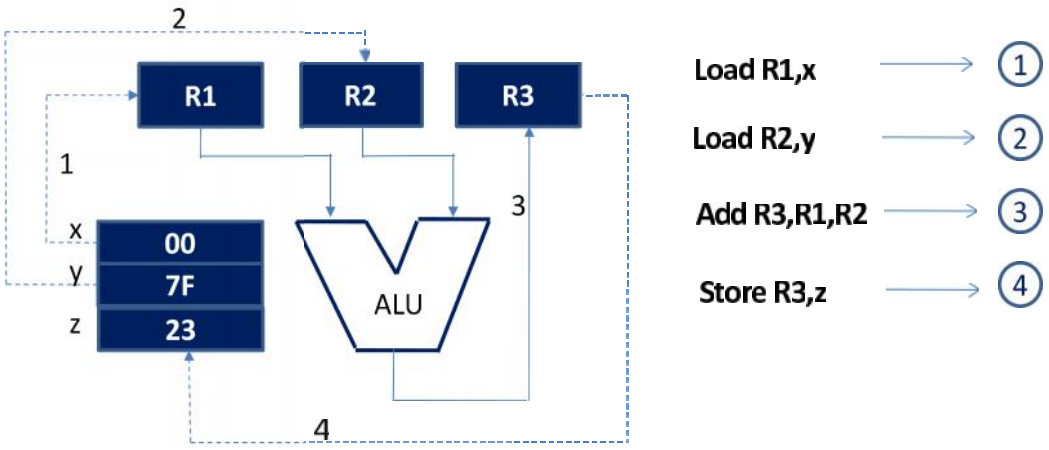

Figure: The concept of load store architecture

- The first instruction Load R1,x loads the register R1 with the content of memory location x.
- The second instruction Load R2,y loads the register R2 with the content of memory location y.
- The third instruction Add R3,R1,R2 adds the content of registers R1 and R2 and stores the result in register R3.
- The fourth instruction Store R3,z stores the content of register R3 in memory location z.

- The conventional instruction execution by the processor follows the **fetch-decode-execute** sequence.
  - Fetch part fetches the instruction from program memory
  - Decode part decodes the instruction to generate the necessary control signals.
  - Execute stage reads the operands, performs ALU operations and stores the result.

- Instruction pipelining refers to the overlapped execution of instructions.
- Under normal program execution flow it is meaningful to fetch the next instruction to execute, while the decoding and execution of the current instruction is in progress.
- If the current instruction in progress is a program control flow transfer instruction like jump or call instruction, then the instruction fetched is flushed and a new instruction fetch is performed to fetch the instruction.

2. Application Specific Integrated Circuits (ASICs)

- ASIC is a microchip designed to perform a specific or unique application.
- It is used as replacement to conventional general purpose logic chips.
- It integrates several functions into a single chip and thereby reduces the system development cost.
- Most of the ASICs are proprietary products.
- As a single-chip, ASIC consumes a very small area in the total system and thereby helps in the design of smaller systems with high capabilities/functionalities.
- ASIC based systems are profitable only for large volume commercial productions.
- Fabrication of ASICs requires a non-refundable initial investment for the process technology and configuration expenses. This investment is known as Non-Recurring Engineering Charges (NRE) and it is a onetime investment.
- If the NRE is borne by a third party and the ASIC is made openly available in the market, the ASIC is referred as Application Specific Standard Product (ASSP).
- The ASSP is marketed to multiple customers just as a general-purpose product is, but to a smaller number of customers since it is for a specific application.

- Since ASICs are proprietary products, the developers of such chips may not be interested in revealing the internal details of it.

3. Programmable Logic Devices (PLDs)

- Logic devices provide specific functions, including device-to-device interfacing, data communication, signal processing, data display, timing and control operations, and almost every other function a system must perform.
- Logic devices can be classified into two broad categories:
  - **Fixed Logic Devices:** As the name indicates, the circuits in a fixed logic device are permanent, they perform one function or set of functions - once manufactured, they cannot be changed.
  - **Programmable Logic Devices:** PLDs offer customers a wide range of logic capacity, features, speed, and voltage characteristics – and these devices can be re-configured to perform any number of functions at any time.
- With programmable logic devices, designers use inexpensive software tools to quickly develop, simulate, and test their designs. Then, a design can be quickly programmed into a device and immediately tested in a live circuit.
- There are no NRE costs and the final design is completed much faster than that of a custom, fixed logic devices.
- Another key benefit of using PLDs is that during the design phase, customers can change the circuitry as often as they want until the design operates to their satisfaction.
- PLDs are based on re-writable memory technology – to change the design, the device is simply reprogrammed. Once the design is final, customers can go into immediate production by simply programming as many PLDs as they need with the final software design file.

- The two major types of PLDs are
  - **Field Programmable Gate Arrays (FPGAs)**
    - FPGAs offer the highest amount of logic density, the most features, and the highest performance.
    - The largest FPGA (Xilinx Virtex) provides eight million "system gates".
    - FPGAs are used in a wide variety of applications ranging from data processing and storage, to instrumentation, telecommunications, and digital signal processing.
    - FPGAs are especially popular for prototyping ASIC designs where the designer can test his design by downloading the design file into an FPGA device. Once the design is set, hardware chips are produced for faster performance.
  - **Complex Programmable Logic Devices (CPLDs)**
    - CPLDs offer much smaller amounts of logic-up to about 10,000 gates.
    - CPLDs offer very predictable timing characteristics and are therefore ideal for critical control applications.
    - CPLDs such as the Xilinx CoolRunner series also require extremely low amounts of power and are very inexpensive, making them ideal for cost-sensitive, battery-operated, portable applications such as mobile phones and digital handheld assistants.

Advantages of PLD:

PLDs offer a number of important advantages over fixed logic devices, including the following:

1. PLDs offer customers much more flexibility during the design cycle because design iterations are simply a matter of changing the programming file, and the results of design changes can be seen immediately in working parts.
2. PLDs do not require long lead times for prototypes or production parts – the PLDs are already on a distributor's shelf and ready for shipment.

3. PLDs do not require customers to pay for large NRE costs – PLD suppliers incur those costs when they design their programmable devices.
4. PLDs allow customers to order just the number of parts they need, when they need them, allowing them to control inventory. Customers who use fixed logic devices often end up with excess inventory which must be scrapped, or if demand for their product surges, they may be caught short of parts and face production delays.
5. PLDs can be reprogrammed even after a piece of equipment is shipped to a customer. PLD equipment manufacturers can add new feature or upgrade products that are already in the field. To do this, they simply upload a new programming file to the PLD, via the Internet, creating new hardware logic in the system.

4. Commercial Off-the-Shelf Components (COTS)

- COTS products are designed in such a way to provide easy integration and interoperability with existing system components.
- The COTS component itself may be developed around a general purpose or domain specific processor or an ASIC or a PLD.
- Typical examples of COTS hardware unit are remote controlled toy car control units including the RF circuitry part, high performance, high bandwidth analog-to-digital converters, devices and components for operation at very high temperatures, UV/IR detectors, etc.
- The major advantage of using COTS is that they are readily available in the market, cheap, and a developer can cut down his/her development time to a great extent.
- The TCP/IP plug-in module available from various manufacturers like WIZnet, Viewtool, etc are very good examples of COTS product.
  - This network plug-in module gives the TCP/IP connectivity to the system you are developing.
- A COTS component manufactured by a vendor need not have hardware plug-in and firmware interface compatibility with one

manufactured by a second vendor for the same application. This restricts the end-user to stick to a particular vendor for particular COTS. This greatly affects the product design.

- The major drawback of using COTS components in embedded design is that the manufacturer of the COTS component may withdraw the product or discontinue the production of the COTS at any time if a rapid change in technology occurs, and this will adversely affect a commercial manufacturer of the embedded system which makes use of the specific COTS product.

Memory

- Memory is an important part of a processor/controller based embedded systems.
- Some of the processors/controllers contain built in memory and this memory is referred as **on-chip memory**.
- Some processors/controllers do not contain any memory inside the chip and requires external memory to be connected with controller/processor. It is called **off-chip memory**.

Program Storage Memory (ROM)

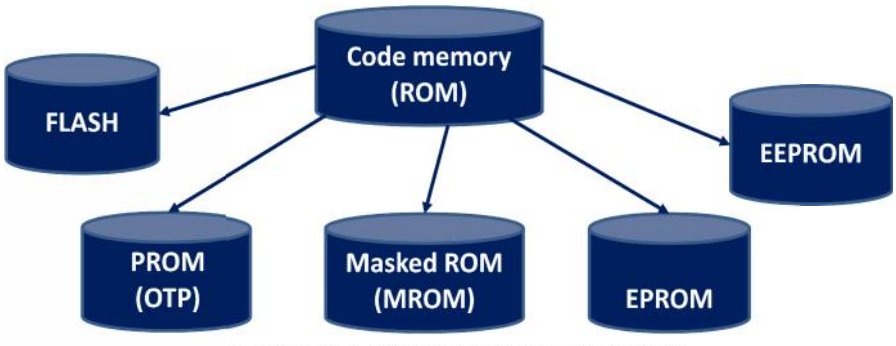

Figure: Classification of Program Memory (ROM)

- The program memory or code storage memory of an embedded system stores the program instructions and it can be classified into different types as per the block diagram given in figure.

- The code memory retains its contents even after the power is turned off. It is generally known as non-volatile storage memory.
- Depending on the fabrication, erasing and programming techniques, program storage memory is classified into the following types.

Masked ROM (MROM):

- MROM is a one-time programmable device.
- It makes use of the hardwired technology for storing data.
- The device is factory programmed at the time of production itself, according to the data provided by the end user.
- The primary advantage of this is low cost for high volume production.
- They are the least expensive type of solid state memory.
- MROM is a good candidate for storing the embedded firmware for low cost embedded devices.
- The limitation with MROM based firmware storage is the inability to modify the device firmware against firmware upgrades.

Programmable ROM (PROM) / (OTP):

- Unlike MROM, One Time programmable Memory (OTP) or PROM is not pre-programmed by the manufacturer. The end user is responsible for programming these devices.
- This memory has ***nichrome*** or ***polysilicon*** wires arranged in a matrix. These wires can be functionally viewed as fuses.
- It is programmed by a PROM programmer which selectively burns the fuses according to the bit pattern to be stored. Fuses which are not blown/burned represent a logic “1” whereas fuses which are blown/burned represents a logic “0”.
- OTP is widely used for commercial production of embedded systems whose prototyped versions are proven and the code is finalized. It is a low cost solution for commercial production.
- OTPs cannot be reprogrammed.

Erasable Programmable ROM (EPROM):

- EPROM gives the flexibility to re-program the same chip.
- EPROM stores the bit information by charging the floating gate of an FET.
- Bit information is stored by using an EPROM programmer, which applies high voltage to charge the floating gate.
- EPROM contains a quartz crystal window for erasing the stored information.
  - If the window is exposed to ultraviolet rays for a fixed duration, the entire memory will be erased.
- Even though the EPROM chip is flexible in terms of re-programmability, it needs to be taken out of the circuit board and put in a UV eraser device for 20 to 30 minutes. So, it is a tedious and time-consuming process.

Electrically Erasable Programmable ROM (EEPROM):

- As the name indicates, the information contained in the EEPROM can be altered by using electrical signals at the register/byte level. They can be erased and reprogrammed in-circuit.
  - These chips include a chip erase mode and in this mode they can be erased in a few milliseconds.
- It provides greater flexibility for system design.
- The only limitation is their capacity is limited when compared with the standard ROM (A few kilobytes).

FLASH:

- FLASH is the latest ROM technology and is the most popular ROM technology used in today's embedded designs.
- FLASH memory is a variation of EEPROM technology. It combines the reprogrammability of EEPROM and the high capacity of standard ROMs.

- FLASH memory is organized as sectors or pages. It stores information in an array of floating gate MOSFET transistors.
- The erasing of memory can be done at sector level or page level without affecting the other sectors or pages.
- Each sector/pages should be erased before re-programming.
- The typical erasable capacity of FLASH is of the order of a few 1000 cycles.

Read-Write Memory / Random Access Memory (RAM)

- RAM is the data memory or working memory of the controller/processor.
- Controller/processor can read from it and write to it.
- RAM is volatile, meaning when the power is turned off, all the contents are destroyed.
- We can access the desired memory location directly without the need for traversing through the entire memory locations to reach the desired memory position.
- RAM generally falls into three categories:
  - Static RAM (SRAM), Dynamic RAM (DRAM) and Non-volatile RAM (NVRAM).

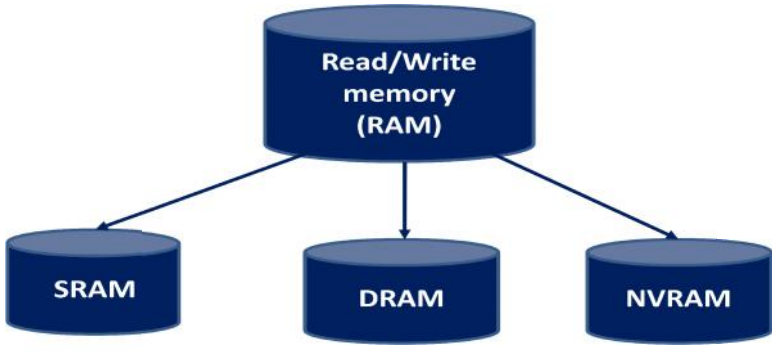

Figure: Classification of Working Memory (RAM)

Static RAM (SRAM):

- SRAM is the fastest form of RAM available.
- SRAM stores data in the form of voltage. They are made up of flip flops.

- In typical implementation, an SRAM cell (bit) is realized using six transistors (or 6 MOSFETs).
- Four of the transistors are used for building the latch part of the memory cell and two for controlling the access.
- In its simplest representation, an SRAM cell can be visualized as shown in below figure:

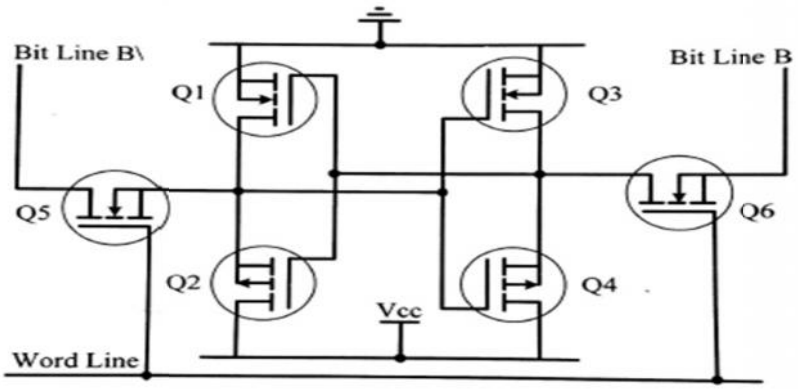

Figure: SRAM cell implementation

- This implementation in its simpler form can be visualized as two-cross coupled inverters with read/write control through transistors. The four transistors in the middle form the cross-coupled inverters. This can be visualized as shown in below figure:

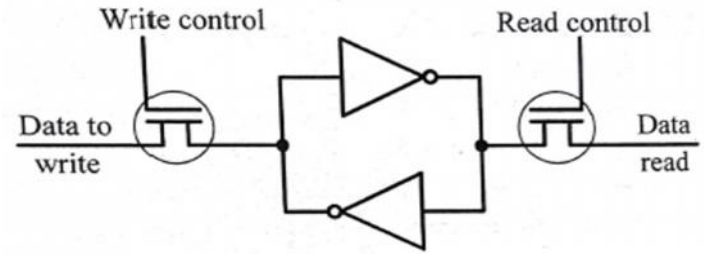

Figure: Visualization of SRAM cell

- From the SRAM implementation diagram, it is clear that access to the memory cell is controlled by the line Word Line, which controls the access transistors (MOSFETs) Q5 and Q6.
- The access transistors control the connection to bit lines B & B\.
- In order to write a value to the memory cell, apply the desired value to the bit control lines (For writing 1, make B=1 and B\=0; For writing 0,

make B=0 and B\=1) and assert the Word Line (Make Word line high). This operation latches the bit written in the flip-flop.

- For reading the content of the memory cell, assert both B and B\ bit lines to 1 and set the Word line to 1.
- The major limitations of SRAM are low capacity and high cost. Since a minimum of six transistors are required to build a single memory cell, imagine how many memory cells we can fabricate on a silicon wafer.

Dynamic RAM (DRAM):

- DRAM stores data in the form of charge. They are made up of MOS transistor gates.
- The advantages of DRAM are its high density and low cost compared to SRAM.
- The disadvantage is that since the information is stored as charge it gets leaked off with time and to prevent this they need to be refreshed periodically.
- Special circuits called DRAM controllers are used for the refreshing operation.
- The refresh operation is done periodically in milliseconds interval.
- The figure illustrates the typical implementation of a DRAM cell.

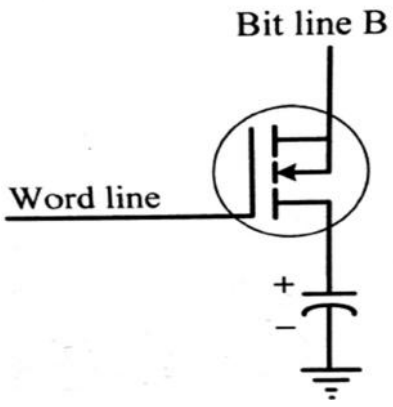

Figure: DRAM cell implementation

- The MOSFET acts as the gate for the incoming and outgoing data whereas the capacitor acts as the bit storage unit.

Differences between SRAM and DRAM

SRAM cell	DRAM cell
Made up of 6 CMOS transistors (MOSFET)	Made up of a MOSFET and a capacitor
Doesn't require refreshing	Requires refreshing
Low capacity (Less dense)	High capacity (Highly dense)
Fast in operation. Typical access time is 10ns	Slow in operation due to refresh requirements. Typical access time is 60ns.

NVRAM:

- Non-volatile RAM is a random access memory with battery backup.
- It contains static RAM based memory and a minute battery for providing supply to the memory in the absence of external power supply.
- The memory and battery are packed together in a single package.
- NVRAM is used for the non-volatile storage of results of operations or for setting up of flags, etc.
- The life span of NVRAM is expected to be around 10 years.
- DS1744 from Dallas is an example for 32KB NVRAM.

Memory according to the type of Interface

- The interface (connection) of memory with the processor /controller can be of various types.
- It may be a parallel interface (The parallel data lines (D0-D7) for an 8-bit processor/controller will be connected to D0-D7 of the memory)
- It may be serial interface
  - I2C (Pronounced as I square C. It is a 2 line serial interface)
  - SPI (Serial Peripheral Interface, $2+n$ line interface where n stands for the total number of SPI bus devices in the system)

- A single wire interconnection)like Dallas 1-wire interface)

- Serial interface is commonly used for data storage memory like EEPROM.
- The memory density of a serial memory is usually expressed in terms of kilobits, whereas that of a parallel interface memory is expressed in terms of kilobytes.

Memory Shadowing

- Generally, the execution of a program or a configuration from ROM is very slow (120 to 200 ns) compared to the execution from a RAM (40 to 70 ns).
- Shadowing is a technique adopted to solve the execution speed problem in processor-based systems.
- In computer systems and video systems there will be a configuration holding ROM called Basic Input Output System (BIOS).
- In personal computer systems BIOS stores the hardware configuration information like the address assigned for various serial port, etc.
    - Usually it is read and the system is configured according to it during system boot up and it is time consuming.
- Now the manufacturers included a RAM behind the logical layer of BIOS at its same address as a shadow to the BIOS and the first step that happens during the boot up is copying the BIOS to the shadowed RAM and write protecting the RAM then disabling the BIOS reading.

Memory Selection for Embedded Systems

- Embedded systems require
    - A program memory for holding the control algorithm or embedded OS and the applications designed to run on top of it
    - Data memory for holding variables and temporary data during task execution
    - Memory for holding non-volatile data (like configuration data, look up table etc) which are modifiable by the application.

- The memory requirement for an embedded system in terms of RAM and ROM is solely dependent on the type of the embedded system and the applications for which it is designed.
- Consider a simple electronic toy design as an example.
  - A microcontroller with few bytes of internal RAM, few bytes or kilobytes of FLASH memory and few bytes of EEPROM can be used for designing the system. There is no need for external memory at all. (A PIC microcontroller can be used)
- If the embedded design is based on an RTOS, the RTOS requires certain amount of RAM for its execution and ROM for storing the RTOS *image* (Binary data generated by the compilation of all RTOS source files).
  - A smart phone with Windows mobile operating system is a typical example for embedded device with OS.
    - Say 512 MB RAM and 1 GB ROM are the minimum requirements for running the Windows mobile device. Extra RAM and ROM is needed for running user applications.

- There are two parameters for representing a memory.
  - The first one is the size of the memory chip
    - Memory chips come in standard sizes like 512 bytes, 1024 bytes (1 KB), 2048 bytes (2 KB), 4 KB, 8 KB, 32 KB, 64 KB, 128 KB, 256 KB, 512 KB, 1024 KB (1 MB), etc.
    - While selecting the memory size, the address range supported by the processor must be known.
  - The second parameter that needs to be considered in selecting a memory is the word size of the memory.
    - The word size refers to the number of memory bits that can be read/written at a time.
    - Ensure that the word size supported by the memory chip matches with the data bus width of the processor/controller.

- FLASH memory is the popular choice for ROM in embedded applications.
- FLASH memory comes in two major variants, namely, NAND and NOR FLASH.
  - NAND FLASH is a high-density low cost non-volatile storage memory.
  - NOR FLASH is less dense and slightly expensive. It supports the Execute in Place (XIP) technique for program execution.

******* **END OF UNIT - 2** *******