

Module III- Introduction to Embedded System

Dr. Rejeesh Rayaroth

Asst. Professor

CSE

MITE

SYLLABUS

Module 3: Introduction to Embedded Systems	No. of Hrs: 9+4
<p>Embedded System Components, Embedded Vs General computing system, Classification of Embedded systems, Applications areas of embedded systems, Core of an Embedded System : processor/controller, Memory, Sensors and Actuators, LED, 7 segment LED display, stepper motor, Keyboard, Push button switch, Communication Interface (onboard and external types), Embedded firmware, Other system components</p>	
<p>Laboratory Components:</p>	
<ol style="list-style-type: none">1. Using Keil software write a program to find the largest or smallest number in an array of 32 numbers2. Using Keil software write a program to arrange a series of 32 bit numbers in ascending/descending order	

WHAT IS AN EMBEDDED SYSTEM ?

- ❑ An Embedded System is an Electronic or Electro-mechanical system designed to perform a specific function.
- ❑ An Embedded System is a combination of hardware and Software(Also called Firmware).

DIFFERENCE BETWEEN GENERAL PURPOSE COMPUTING SYSTEM AND EMBEDDED SYSTEM

General Purpose Computing System

- Combination of general h/w and general purpose operating system for executing the Appln.
- Contains General purpose OS.
- Applications are alterable by the user.
- Performance is the key factor in selecting the system.
- Response time are not Critical
- Need not be deterministic execution behavior.

Embedded System

- Combination of special purpose hardware and embedded OS for executing , specific set of Appln.
- May or May not contain the OS.
- Firmware of the Embedded Systems is pre-programmed and is not alterable by end users.
- Application specific requirements (performance, power requirements, memory, usage etc.) are key deciding factors
- Mission Critical Applications.
- Execution behavior is deterministic

Classification of Embedded System

Embedded Systems can be classified on the basis following:

- ❑ Based on Generations.**
- ❑ Based on Complexity and performance Requirements.**
- ❑ Based on Deterministic Behavior.**
- ❑ Based on Triggering**

CLASSIFICATION ON THE BASIS OF GENERATIONS

□ Classification based on Generations

- ❖ **First Generation**
- ❖ **Second Generation**
- ❖ **Third Generation**
- ❖ **Fourth Generation**

FIRST GENERATION & SECOND GENERATION

First Generation

- ❑ Simple Hardware
- ❑ uses 8 bit Microprocessor and 4 bit Microcontrollers
- ❑ Make use of Assembly code.
- ❑ **Ex: Digital telephone keypads, Stepper motor control**

Second Generation

- ❑ Embedded System built around 16 bit microprocessor and 8 bit or 16 bit micro-controllers.
- ❑ Powerful Complex Instruction set than first generation.
- ❑ Requires Embedded OS for their operations.
- ❑ **Ex: Data Acquisition system, SCADA Systems (Supervisory Control and Data Acquisition)**

Third Generations

Third Generations

- ❑ Makes use of powerful processor like 32 bit microprocessor.
- ❑ Domain Specific processor/ controller like Digital signal Processor(DSP) and Application specific Integrated Circuit(ASIC) were introduced.
- ❑ More complex instruction sets with features of pipelining were introduced.
- ❑ Dedicated Real Time OS and General purpose OS were introduced.
- ❑ **Ex: Robotics, Networking, Industrial process, media**

FOURTH GENERATIONS

Fourth Generations

- ❑ Introduction of **System on chip** (SoC) reconfigurable processors, brings high performance, miniaturization into the Embedded Market.
- ❑ Application to be embedded on the single chip.
- ❑ Used High-Performance Real time Embedded OS for their functioning.
- ❑ Ex: **Smart phone devices, Mobile Internet devices.**

CLASSIFICATION BASED ON COMPLEXITY AND PERFORMANCE.

Classification based on Complexity and Performance.

on the basis of Complexity & performance Embedded Systems are classified as

- Small Scale Embedded System
- Medium Scale Embedded System
- Large Scale Embedded Systems/Complex systems.

SMALL SCALE EMBEDDED SYSTEMS

Small Scale Embedded Systems:

- Involves the applications which are simple.
- Performance requirements are not critical
- Low-performance and Low cost 8/16 bit microprocessor.
- May or May not contain the OS.
- Ex: Electronic Toys,**

MEDIUM SCALE EMBEDDED SYSTEMS

Medium Scale Embedded Systems

- ❑ Embedded System which are slightly complex in hardware and firmware requirements.
- ❑ Built Around Medium performance, involves 16/32 bit $\mu\text{P}/\mu\text{C}$ or **Digital Signal Processor(DSP)**.
- ❑ Contains **Embedded OS** or **General purpose OS** for their functioning.

LARGE-SCALE EMBEDDED SYSTEM/ COMPLEX SYSTEMS:

Large Scale Embedded System/ Complex systems:

- Involves complex hardware and firmware.
- High Performance, Mission Critical Applications.
- Built using high performance 32-bit or 64-bit $\mu\text{P}/\mu\text{C}$, SoC and multi-core processors and programmable logic devices.
- Decoding/ Encoding of media, cryptographic function requirements which are done using Hardware Accelerator.
- Contains high performance RTOS for task Scheduling, prioritization and Management.

CLASSIFICATION BASED ON DETERMINISTIC BEHAVIOR

- ❑ 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**.
- ❑ We already discussed about **hard and soft real time** systems in first module.

CLASSIFICATION BASED ON TRIGGERING

- ❑ 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**.
- ❑ This will be Discussed later

MAJOR APPLICATION AREAS OF EMBEDDED SYSTEMS:

□ Major Application Areas of Embedded Systems:

□ *Consumer Electronics:*

- ❖ Camcorders
- ❖ Cameras etc

□ *Household Appliances:*

- ❖ TV,
- ❖ DVD Players,
- ❖ refrigerators,
- ❖ washing machine,
- ❖ Microwave Oven.

Major Application Areas of Embedded Systems Ctd.

❑ *Home Automation and Security Systems:*

- ❖ Air conditioners,
- ❖ sprinklers,
- ❖ intruder detection system, CCTV,
- ❖ fire Alarms.

❑ *Automotive Industry:*

- ❖ Anti-lock Braking System (ABS),
- ❖ Engine Control, Ignitions systems,
- ❖ Navigation Systems etc.

❑ *Telecom:*

- ❖ Cellular Telephones,
- ❖ Telephone switches,
- ❖ handset multimedia Applications.

Major Application Areas of Embedded Systems Ctd.

❑ *Computer Peripherals:*

- ❖ Printers, Scanners,
- ❖ Fax Machines etc

❑ *Computer Networking Systems:*

- ❖ Network Routers,
- ❖ switches,
- ❖ handset multimedia Applications.

❑ *Healthcare:*

- ❖ Different Kinds of Scanners,
- ❖ EEG, ECG, MRI etc

❑ *Measurement and Instrumentation:*

- ❖ Digital multimeters,
- ❖ Digital CRO,
- ❖ logic Analyzers
- ❖ PLC Systems etc..

❑ *Banking and Retail:*

- ❖ ATM,
- ❖ currency counters.

❑ *Card Readers:*

- ❖ Bar Code,
- ❖ Smart card Readers,
- ❖ Hand held Devices etc..

❑ *Wearable device:*

- ❖ Health and fitness trackers,
- ❖ Smartphone screen Extension for notifications.

❑ Cloud Computing and Internet of Things(IOT)

PURPOSE OF EMBEDDED SYSTEMS:

□ All Embedded systems are designed to serve the purpose of one or more 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.

Data Collection/ Storage/ Representation:

- ❑ These type of Embedded System(ES) performs Access of data from the External World.
- ❑ Data Collection done for the purpose of storage, analysis, manipulation and transmission.
- ❑ Data refers to Text, Voice, Image, video, Electrical Signals and other measurable quantities.

-
- ❑ ES with a analog capturing techniques collect data directly in the form of the analog signal.
 - ❑ ES with digital data , will first convert the analog signal to digital data using A/D Converter and then it will store.
 - ❑ ES are used to collect the data, and with the collected data they are going to make the analysis. ex: Medical Applications

-
- ❑ ES, which are generally used for measurement application without storage will collect the data and represent them by means of the graphical representation.
 - ❑ ex: Digital CRO, Analog CRO
 - ❑ Digital Camera is the typical example of ES used for data collection/storage and representation.

DATA COMMUNICATION:

- ❑ ES used in data communication like complex satellite communication system to simple home networking system.
- ❑ In this, data collected at one terminal has to be transferred to the other terminal located in the remote location.
- ❑ Transmission medium can be either wired or wireless. Ex: Wireless Router, modem, Bluetooth.

Data collecting embedded system can incorporate the data communication unit.

Routers, switches are dedicated data transmission ES.

DATA PROCESSING:

- ❑ Data(Voice, image, video, electrical signals or other measurable quantities) collected by an ES can be processed to get a useful information for the purpose of analysis.
- ❑ Used for Application of Speech Processing, Audio video codec, transmission Applications.
- ❑ Ex: Digital hearing aid (It improves the hearing capacity of the impaired persons).

Monitoring:

- ❑ ES designed under this category are used for only for the purpose of monitoring.
- ❑ Used to determine the state of some variables using input sensors.
- ❑ Ex: ECG (Electro Cardio Gram) used for monitoring the heartbeat of a patient. Sensors used are the electrodes connected to the body of the patient.
- ❑ Ex: Digital CRO, Digital multimeters, Logic Analyzers.
- ❑ **Limitations:** They don't have control over the variables.

CONTROL

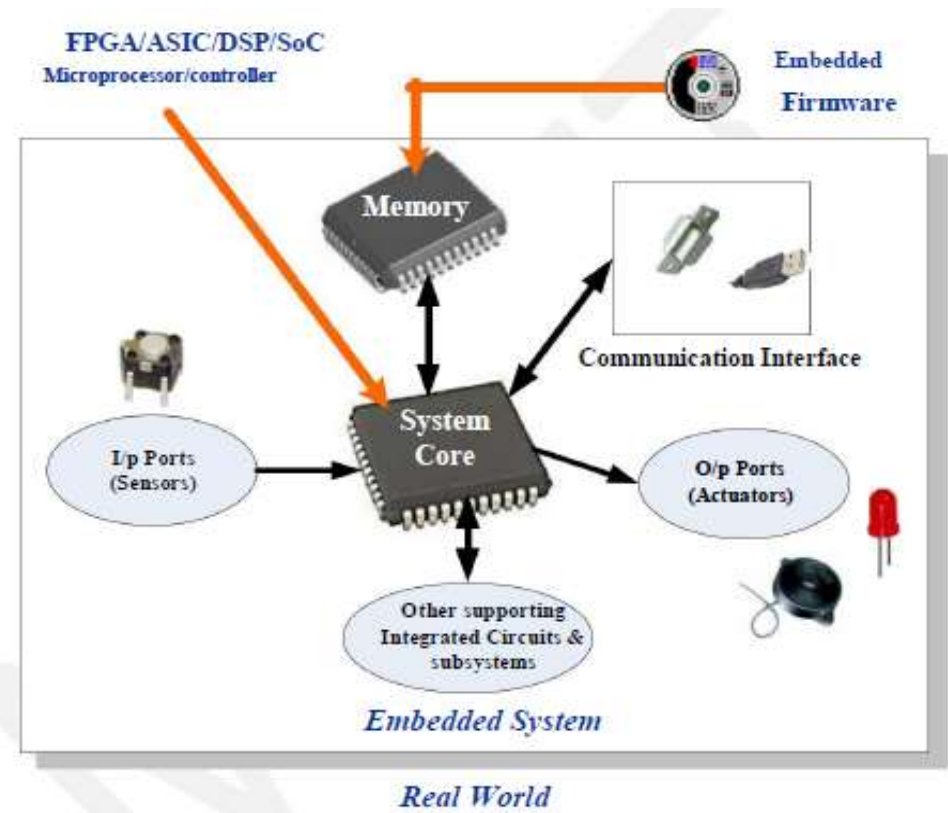
- ❑ ES which has the control over the input variables, in turn the other variables are controlled(o/p variables are controlled).
- ❑ A system with control functionalities will have the sensors and the Actuators.
- ❑ Actuators connected to the output port are controlled according to the changes in input variable to put the impact on the controlling variable.
- ❑ Ex: **Air Conditioner System**: Control the Room temperature to a specified limit.

Application specific user Interface:

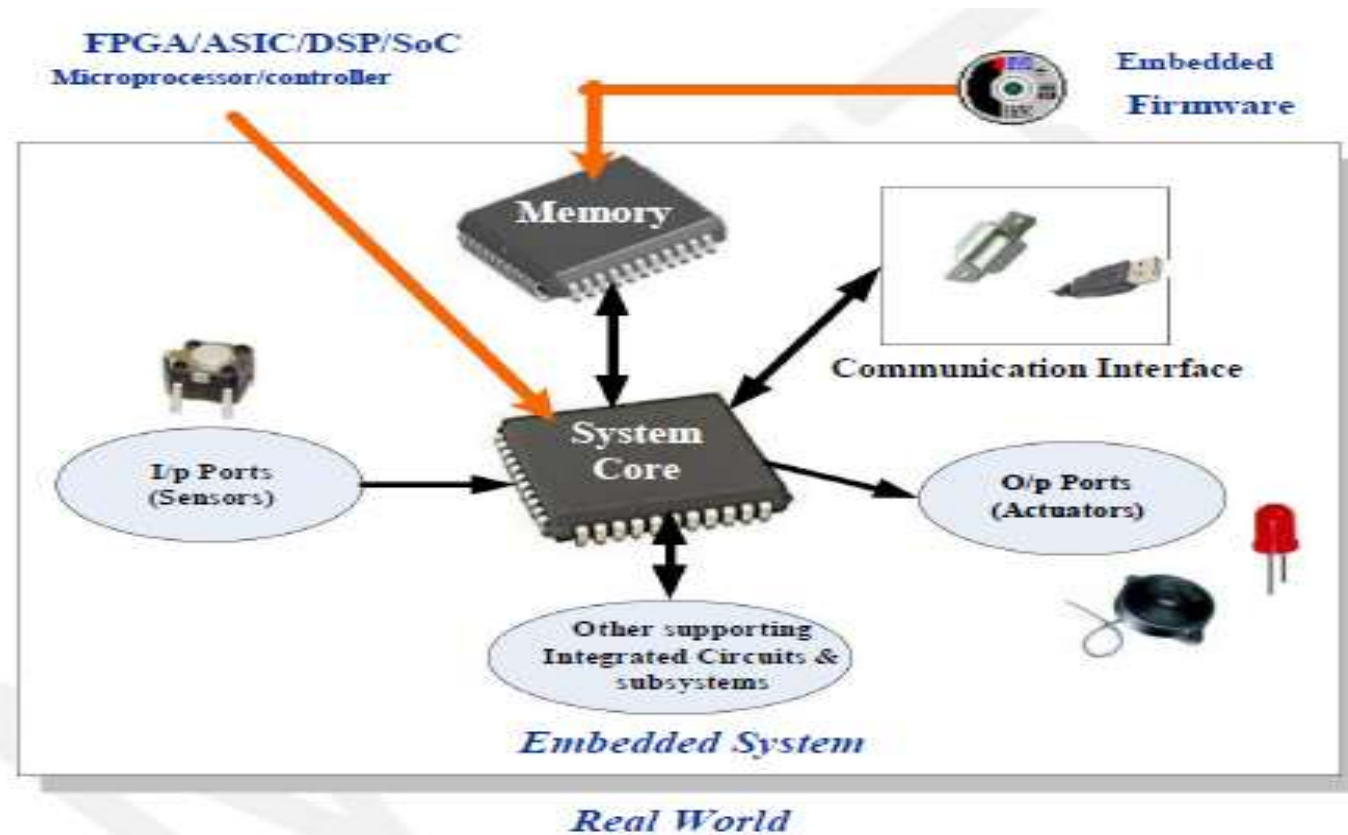
- ❑ Embedded System with the Application specific user interface like buttons, switches, keypad, lights, bell etc.
- ❑ **Ex: Mobile Phone.**
- ❑ Mobile phone the user interface is provided through *Keypads, Graphic LCD Module, system speaker, vibration alert etc..*
- ❑ **Wearable devices:** It refer to the embedded device which are incorporated into accessories and apparels, which we are going to use in our day today life activities

What is an Embedded System ?

- ❑ An Embedded system is a combination of 3 things, Hardware, Software and Mechanical Components and it is supposed to do one specific task only.
- ❑ A typical Embedded system contains a *single chip controller* which acts as the *master brain of the system*.
- ❑ Diagrammatically an Embedded system can be represented as follows:



Elements of an Embedded System



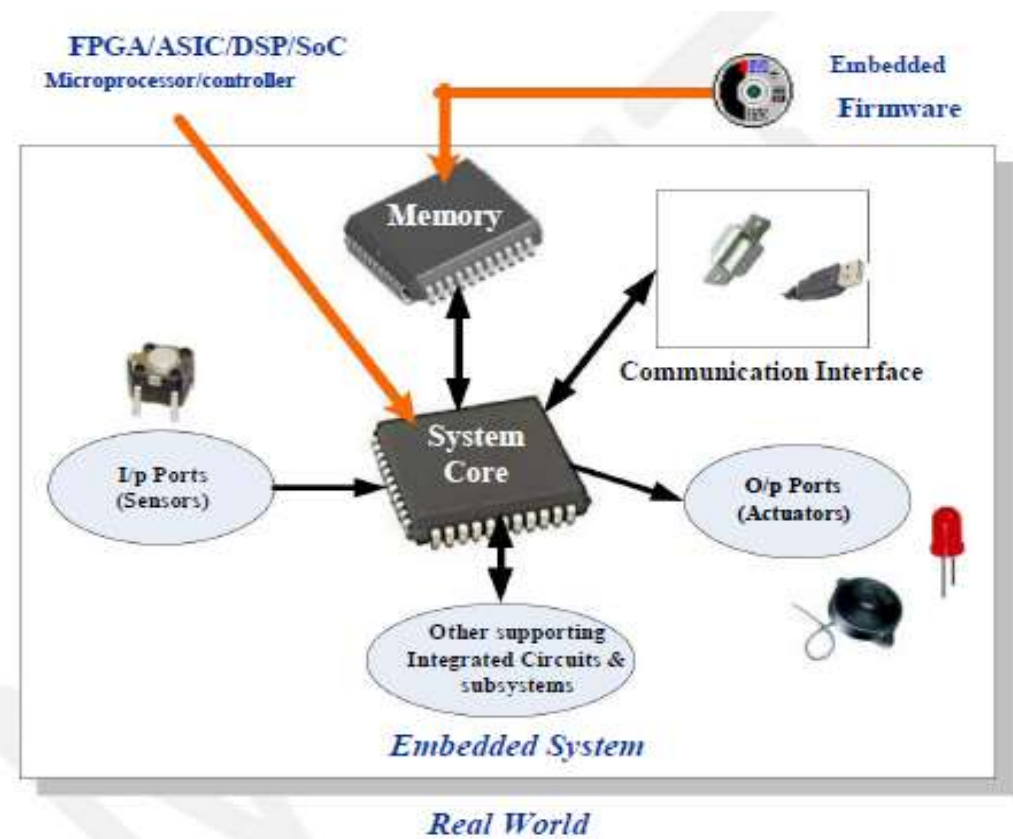
❑ Embedded systems are basically designed to *regulate as physical variable* (such as Microwave Oven) or to *manipulate the state of some devices* by sending some signals to the actuators or devices connected to the output of the system (such as temperature in Air Conditioner), in response to the input signal provided by the end users or sensors which are connected to the input ports.

❑ The control is achieved by processing the information coming from the sensors and user interfaces and controlling some actuators that regulate the physical variable.

-
- ❑ Keyboards, push button, switches are some of the user interface used and the type of the user interface vary from Application to Application.
 - ❑ Embedded System does not require the human intervention but works with the input from the sensors and produce the output with the help of the Actuators.
 - ❑ Memory is used for holding the Code.
 - ❑ Fixed Memory (ROM): Used to hold the code or Program or for holding the control Algorithm.
 - ❑ Most Commonly used Memory for Controlling Algorithm storage are OTP, PROM,EEPROM, FLASH.
 - ❑ In a controller based embedded system, the controller may contain internal memory for storing code such controllers are called Micro-controllers with on-chip ROM, eg. Atmel AT89C51.

The Elements of the Embedded System are comprised of

- Single Chip Controller (Acts as the brain of the system)
- Memory (Embedded Firmware)
- Input ports (Sensors)
- Output Ports (Actuators)
- Communication Interface
- Other Supporting IC's and Subsystem:



Core of an Embedded System

- ❑ Embedded Device are Domain or Application specific device which are built under central core.
- ❑ The Core of an ES falls under any of the following categories:
 - 1. General purpose and Domain Specific Processors**
 1. Microprocessors.
 2. Microcontrollers.
 3. Digital Signal Processors.
 - 2. Application specific IC.**
 - 3. Programmable logic devices.**
 - 4. Commercial off the shelf components(COTS).**

General Purpose and Domain Specific Processors

- ❑ **Microprocessors**
- ❑ **General Purpose Processor(GPP) or Application specific Instruction set Processor (ASIP).**
- ❑ **Microcontrollers**
- ❑ **Microprocessors vs Micro Controllers**
- ❑ **Digital Signal Processors**
- ❑ **RISC vs. CISC Processors/Controllers**
- ❑ **Harvard vs. Von-Neumann Processor/ Controller Architecture**
- ❑ **Big – Endian vs. Little Endian Processors/Controllers.**
- ❑ **Load store Operation and Instruction Pipelining.**

Microprocessors

- ❑ Almost 80% of the embedded systems are processor/ controller based.
- ❑ The processor may be **microprocessor** or a **microcontroller** or **digital signal processor**, depending on the domain and application.
- ❑ **Microprocessor:**
- ❑ It 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.
- ❑ CPU contains the Arithmetic and Logic Unit (ALU), Control Unit and Working registers.

❑ Microprocessor is a dependant unit and it requires the combination of other hardware like Memory, Timer Unit, and Interrupt Controller etc for proper functioning.

❑ Intel claims the credit for developing the first Microprocessor unit Intel 4004, a 4 bit processor which was released in Nov 1971

❑ **Developers of microprocessors.**

❖ Intel – Intel 4004 – November 1971(4-bit)

❖ Intel – Intel 4040.

❖ Intel – Intel 8008 – April 1972.

❖ Intel – Intel 8080 – April 1974(8-bit).

❖ Motorola – Motorola 6800.

❖ Intel – Intel 8085 – 1976.

❖ Zilog - Z80 – July 1976

GENERAL PURPOSE PROCESSOR (GPP) VS APPLICATION SPECIFIC INSTRUCTION SET PROCESSOR (ASIP).

General Purpose Processor (GPP)

- ❑ The Processor designed for general computing tasks.
- ❑ Used for general market, where the volume required is high, due to which the cost will be low compared to ASIP.
- ❑ GPP contains the ALU(Arithmetic and logic unit) and Control Unit(CU).

Application Specific Instruction set Processor (ASIP).

- ❑ ASIP are processor with architecture and instruction set optimized for the specified purpose like networking, telecom, automotive , media Applications and Control Applications.
- ❑ Need for ASIP arises when GPP is unable to meet the user requirement.
- ❑ Micro-controllers, DSP's, System on Chip are example for ASIP.

Microcontrollers

- ❑ A highly integrated silicon chip containing a CPU, scratch pad RAM, Special and General purpose Register Arrays, On Chip ROM/FLASH memory for programs storage , Timer and Interrupt control units and dedicated I/O ports.
- ❑ Microcontrollers can be considered as a super set of Microprocessors.
- ❑ Microcontroller can be general purpose (like Intel 8051, designed for generic applications and domains)
 - ❑ or
 - ❑ application specific (Like Automotive AVR from Atmel Corporation. Designed specifically for automotive applications)

-
- ❑ A microcontroller contains all the necessary functional blocks for independent, working, they found greater place in the embedded domain in place of microprocessors
 - ❑ Microcontrollers are cheap, cost effective and are readily available in the market.
 - ❑ Texas Instruments TMS 1000 is considered as the world's first microcontroller.

MICRO PROCESSOR VS MICROCONTROLLER

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 pre-defined set of Instructions	A microcontroller is a highly integrated chip that contains a CPU, scratch pad 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
Targeted for high end market where performance is important	Targeted for embedded market where performance is not so critical (At present this demarcation is invalid)
Limited power saving options compared to microcontrollers	Includes lot of power saving features

Digital Signal Processors (DSP)

- ❑ They are powerful special purpose 8/16/32 bit μ P designed to meet the computational demands
- ❑ They are 2 to 3 times faster than the GPP in signal processing Applications.
- ❑ DSP's implements the Algorithm(Algo.) in the Hardware, where it speedup the execution, whereas GPP implement the Algo. in the firmware, where the execution depends on the clock of the Processors.
- ❑ DSP's are microchip designed for performing high speed computational operations for **Addition** , **Subtraction**, **Division** and **Multiplication**.

THE DIFFERENT UNITS OF DSP'S

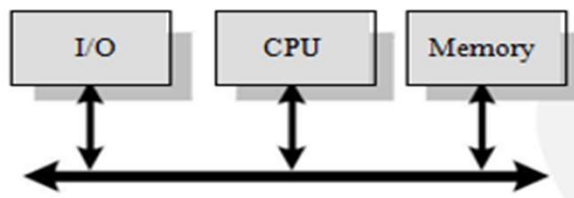
- ❑ The different units of DSP's are **Program Memory, Data Memory, Computational Engine and I/O Unit.**
- ❑ **Program Memory:** It is used to store the data required for processing the data.
- ❑ **Data Memory:** Memory used for Storing temporary variables and data/signal to be processed.
- ❑ **Computational Engine:** Performs the Signal Processing in accordance with the stored program memory.(IT includes many Arithmetic unit which runs parallelly to increase the execution speed.
- ❑ **I/O Unit:** Used to captures signal to be processed and deliver the processed Signals.

RISC vs CISC Processor/controllers

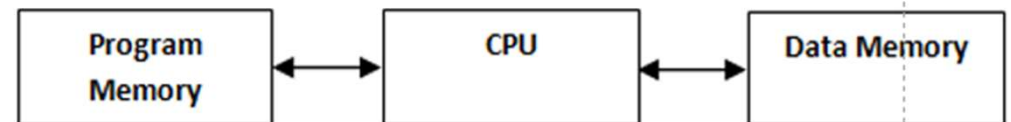
RISC	CISC
Lesser no. of instructions	Greater no. of Instructions
Instruction Pipelining and increased execution speed	Generally no instruction pipelining feature
Orthogonal Instruction Set (Allows each instruction to operate on any register and use any addressing mode)	Non Orthogonal Instruction Set (All instructions are not allowed to operate on any register and use any addressing mode. It is instruction specific)
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
Large number of registers are available	Limited no. of general purpose registers
Programmer needs to write more code to execute a task since the instructions are simpler ones	. A programmer can achieve the desired functionality with a single instruction which in turn provides the effect of using more simpler single instructions in RISC
Single, Fixed length Instructions	Variable length Instructions
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

Harvard or Von-Neumann Processor

- ❑ The terms Harvard and Von-Neumann refers to the processor architecture design.
- ❑ **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



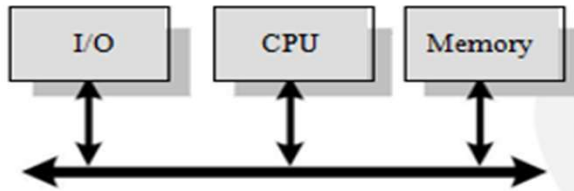
Von - Neumann Architecture



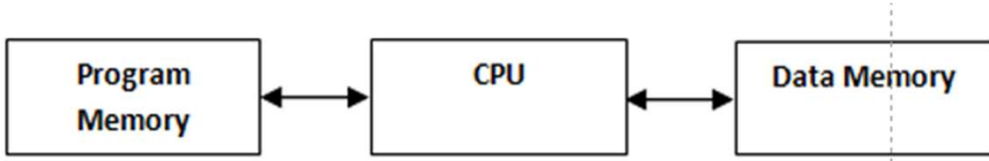
Harvard Architecture

Microprocessors/controllers based on the Harvard architecture

- ❖ will have separate data bus and instruction bus
- ❖ which 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 (“Pre-fetching”).



Von - Neumann Architecture



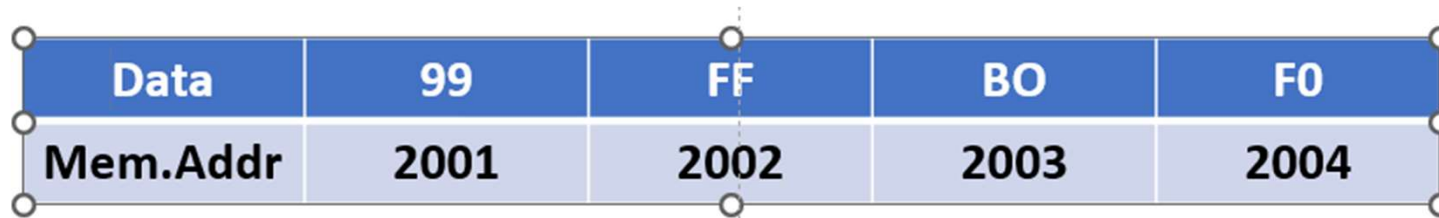
Harvard Architecture

Difference 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
No memory alignment problems	Allows self modifying codes [†]
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 same chip, chances for accidental corruption of program memory

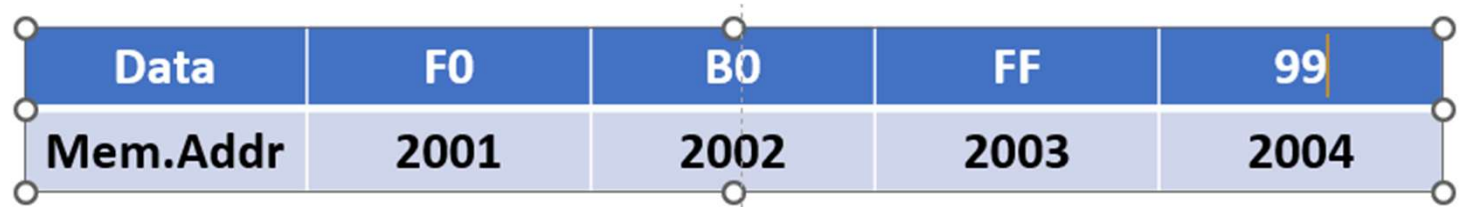
Big Endian and Little Endian Processor/Controllers

- ❑ Endianness specifies the order in which the data is stored in the memory. (Storing data more than one byte)
- ❑ For word length more than one byte the data can be stored in memory in 2 different ways
 - ❑ Little Endian
 - ❑ Big Endian
- ❑ **Little Endian**: Higher order of Data byte at the higher memory and lower order of data byte at the lower memory.
- ❑ Ex: For byte long integer – **F0B0FF99** Byte3Byte2Byte1Byte0



❑ **BIG Endian**: Higher order of Data byte at the lower memory and lower order of data byte at the higher memory.

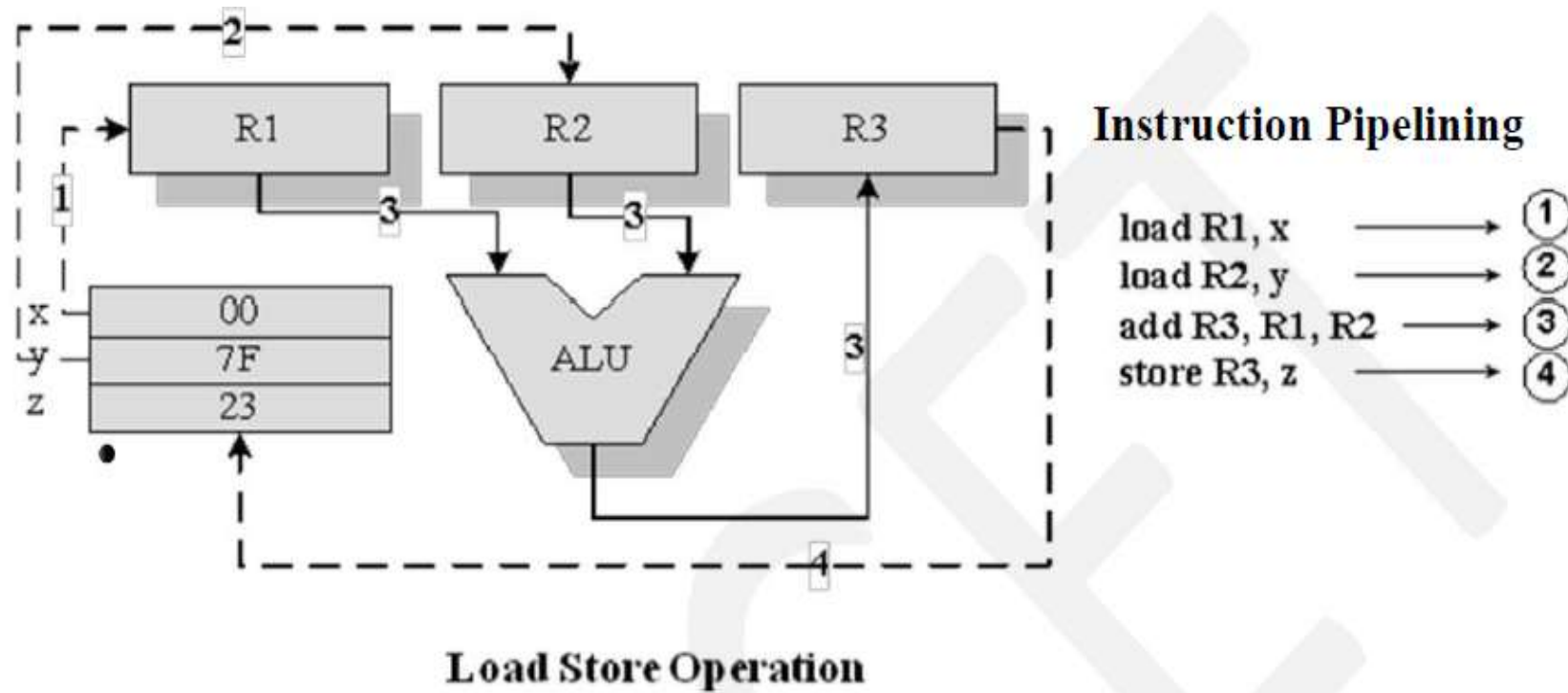
❑ Ex: For byte long integer – F0B0FF99 Byte3Byte2Byte1Byte0



LOAD STORE OPERATION AND INSTRUCTION PIPELINING

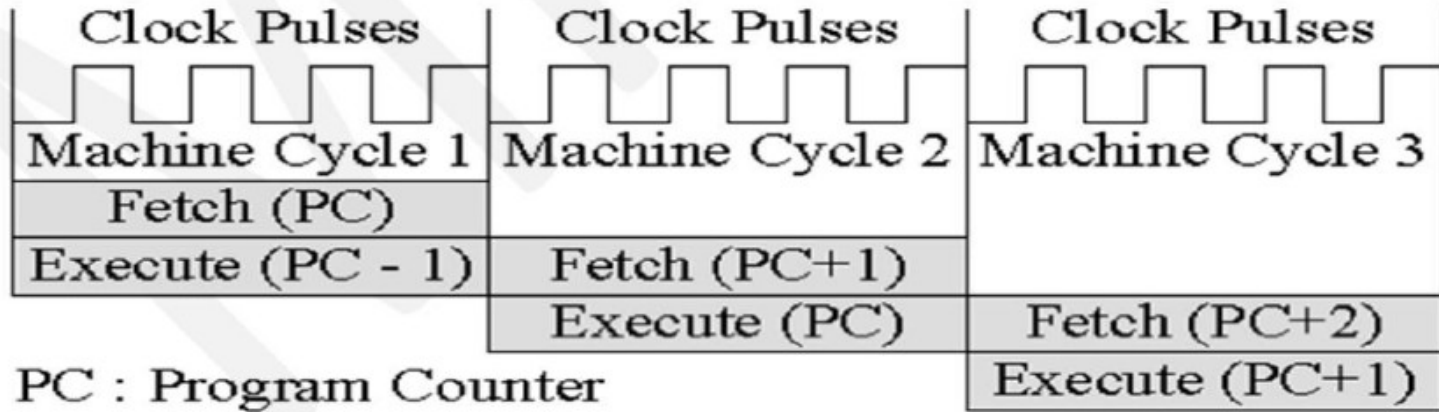
- ❑ RISC Processor Instruction operates on the Registers.
- ❑ The Memory Related Instructions are performed using the Load and store instructions.
- ❑ Load instruction is going to load the data to the register from the memory locations.
- ❑ Store instruction is going to load the data from register to the memory locations.
- ❑ These type of Architecture is referred as Load/Store Architecture

LOAD STORE ARCHITECTURE



-
- ❑ The conventional instruction execution by the processor follows the fetch-decode-execute sequence.
 - ❑ The fetch part fetches the instruction from program memory or code memory and the decode part decodes the instruction to generate the necessary control signals.
 - ❑ The execute stage reads the operands, perform ALU operations and stores the result.

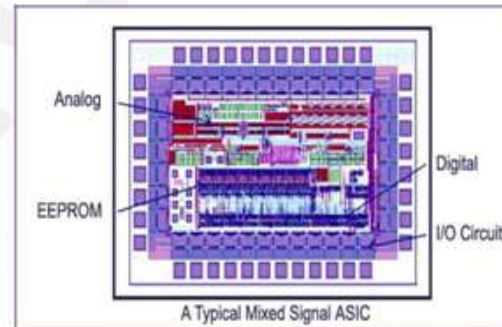
- ❑ During the decode operation, the memory address bus is available and if it is possible to effectively utilize it for an instruction fetch, the processing speed can be increased
- ❑ Instruction pipelining refers to the overlapped execution of instructions



The Single stage pipelining concept

APPLICATION SPECIFIC INTEGRATED CIRCUITS

- ❑ A microchip designed to perform a specific or unique application. It is used as replacement to conventional general purpose logic chips.
- ❑ ASIC integrates several functions into a single chip and thereby reduces the system development cost.
- ❑ As a single chip, ASIC consumes very small area in the total system and thereby helps in the design of smaller systems with high capabilities/functionalities.
- Some ASICs are proprietary products, and the developers are not interested in revealing the internal details.



PROGRAMMABLE LOGIC DEVICES

- ❑ Logic devices provide specific functions as follows
 - ❖ Device-to-device interfacing
 - ❖ Data communication
 - ❖ Signal processing
 - ❖ Data display, Timing and control operations

- ❑ Logic devices can be classified into two broad categories
 - ❖ Fixed
 - ❖ Programmable.

- ❑ 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 characteristic and these devices can be *re-configured to perform any number of functions at any time.*
 - ❑ Designers can use inexpensive software tools to quickly develop, simulate, and test their logic designs in PLD based design.
 - ❑ The design can be quickly programmed into a device, and immediately tested in a live circuit

PROGRAMMABLE LOGIC DEVICES(PLDS): CPLDS AND FPGA

- The 2 types of Programmable Logic devices are
 - ❖ Complex Programmable logic devices
 - ❖ Field Programmable Gate Arrays.

FIELD PROGRAMMABLE GATE ARRAYS:

- ❑ FPGA is an IC designed to be configured by a designer after manufacturing.
- ❑ FPGAs offer the highest amount of logic density, the most features, and the highest performance.
- ❑ Logic gate is Medium to high density ranging from 1K to 500K system gates
- ❑ FPGA devices offer features that support for many of the latest, very fast device-to-device signaling technologies
- ❑ FPGAs are used in a wide variety of applications ranging from data processing and storage, to instrumentation, telecommunications, and digital signal processing

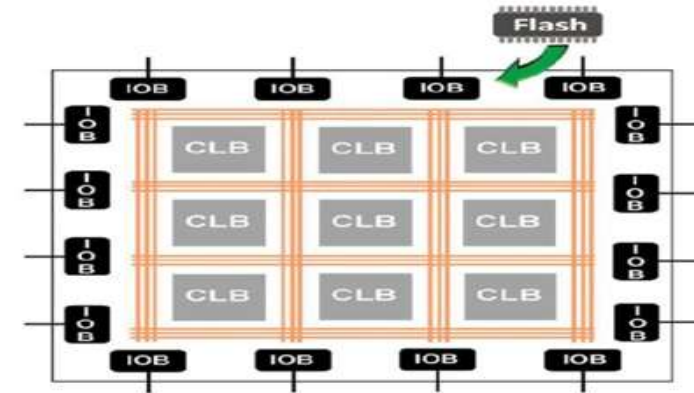
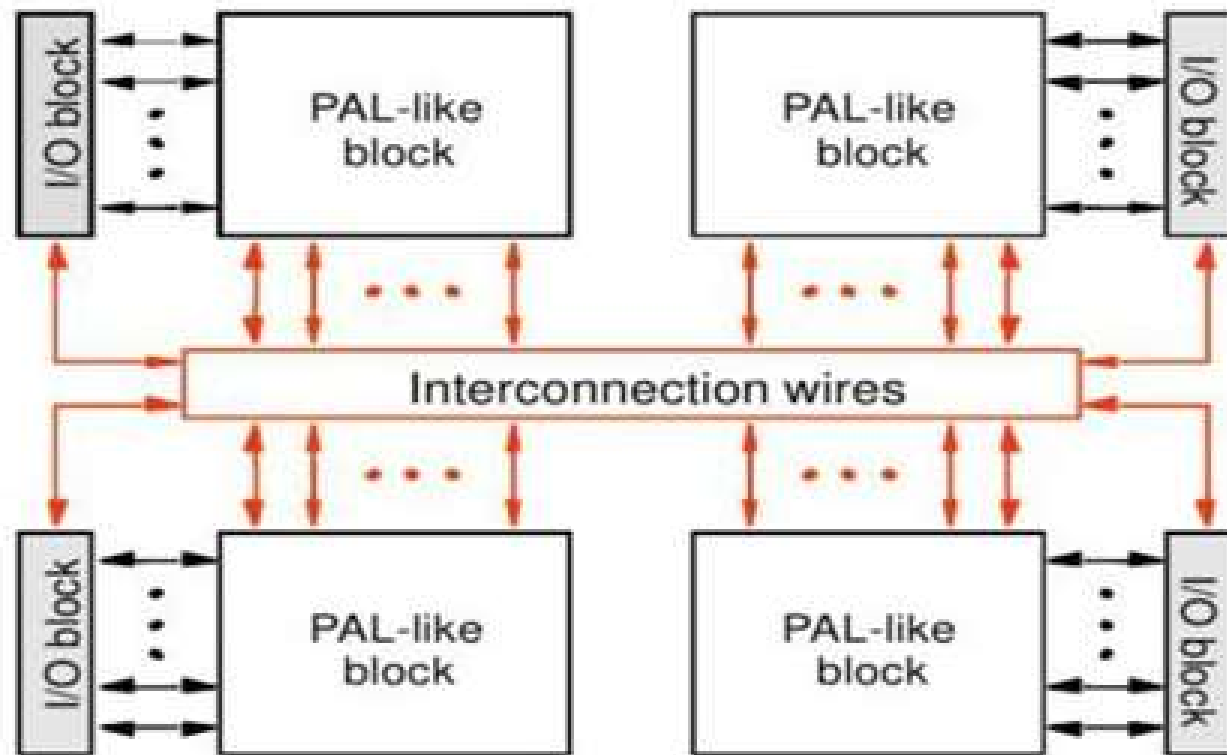


Figure: FPGA Architecture

COMPLEX PROGRAMMABLE LOGIC DEVICES (CPLD)

- ❑ A complex programmable logic device (CPLD) is a programmable logic device with complexity lies between that of PALs and FPGAs.
- ❑ CPLDs, by contrast, offer much smaller amounts of logic – upto 10,000 gates.
- ❑ CPLDs offer very predictable timing characteristics and are therefore ideal for critical control applications.

COMPLEX PROGRAMMABLE LOGIC DEVICES (CPLD)



Structure of a CPLD

ADVANTAGES OF PROGRAMMABLE LOGIC DEVICES

- ❑ PLDs offer customer much more flexibility during design cycle
- ❑ PLDSs do not require long lead times for production-the PLDs are already on a distributor's self and ready for shipment
- ❑ PLDs do not require customers to pay for large costs and purchase expensive mask sets
- ❑ PLDs allow customers to order just the number of parts required when they need them.
- ❑ PLDs are reprogrammable even after a piece of equipment is shipped to a customer.

COMMERCIAL OFF THE SHELF COMPONENT (COTS)

- ❑ A Commercial off-the-shelf (COTS) product is one which is used "as it is".
- ❑ COTS products are designed in such a way to provide easy integration and interoperability with existing system components.
- ❑ Examples: *Remote Controlled Toy Car control unit*
- ❑ It includes
 - ❖ RF Circuitry part,
 - ❖ High performance, high frequency microwave electronics (2 to 200 GHz),
 - ❖ High bandwidth analog-to-digital converters,
 - ❖ Devices and components for operation at very high temperatures,
 - ❖ Electro-optic IR imaging arrays,
 - ❖ UV/IR Detectors etc

-
- ❑ A COTS component in turn contains a General Purpose Processor (GPP) or Application Specific Instruction Set Processor
 - ❑ 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 extend.
 - ❑ There is no need to design the module yourself and write the firmware .
 - ❑ Everything will be readily supplied by the COTs manufacturer.



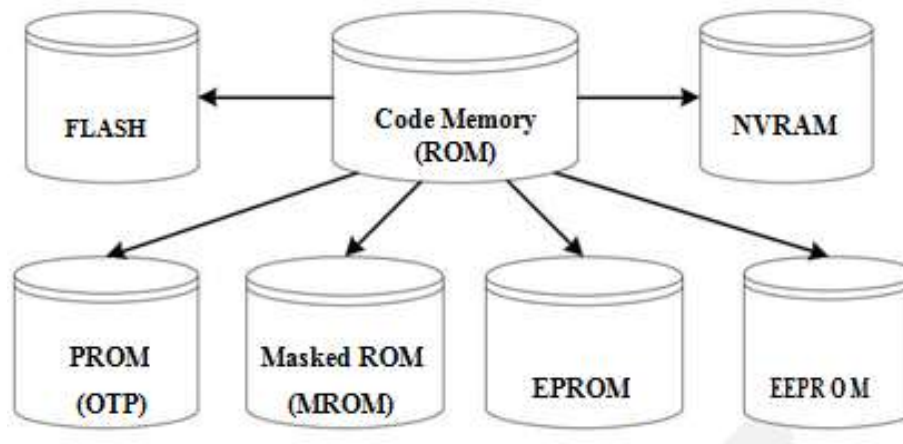
- The major drawback of using COTs component in embedded design is that the manufacturer may withdraw the product or discontinue the production of the COTs at any time if rapid change in technology
- This problem adversely affect a commercial manufacturer of the embedded system which makes use of the specific COTs.

MEMORY

- ❑ Memory is an important part of an embedded system.
- ❑ The memory used in embedded system can be either Program Storage Memory (ROM) or Data memory (RAM).
- ❑ Embedded processors/controllers having built in program memory and data memory is known as *on-chip memory*
- ❑ Embedded processors/controllers do not contain sufficient memory inside the chip and requires external memory called *off- chip memory* or *external memory*.

PROGRAM STORAGE MEMORY (ROM)

- ❑ Stores the program instructions.
- ❑ Retains its contents even after the power to it is turned off.
- ❑ Non volatile storage memory
- ❑ Depending on the fabrication, erasing and programming techniques , they are classified as follows



MASKED ROM (MROM)

Masked ROM (MROM):

- One-time programmable memory.
- Uses hardwired technology for storing data.
- The device is factory programmed by masking and metallization process according to the data provided by the end user.

Advantage of MROM

- low cost for high volume production.
- least expensive type of solid state memory.

□ Different mechanisms are used for the masking process of the ROM

1. Creation of an enhancement or depletion mode transistor through channel implant.
2. By creating the memory cell either using a standard transistor or a high threshold transistor.
 - In the high threshold mode, the supply voltage required to turn ON the transistor.
 - This ensures that the transistor is always off and the memory cell stores always logic 0.

Programmable Read Only Memory (PROM)/ OTP

- ❑ It is not pre-programmed by the manufacturer
- ❑ The end user is responsible for Programming these devices.
- ❑ 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** represents a logic “1” where as **fuses** which are **blown/burned** represents a logic “0”.
 - ❖ The default state is logic “1”.
- ❑ OTP is widely used for commercial production of embedded systems whose proto-typed versions are proven and the code is finalized.

ERASABLE PROGRAMMABLE READ ONLY MEMORY (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 Ultra violet 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 needs to be put in a UV eraser device for 20 to 30 minutes

ELECTRICALLY ERASABLE PROGRAMMABLE READ ONLY MEMORY (EEPROM)

- ❑ **Electrically Erasable Programmable Read Only (EPROM)** memory gives the flexibility to re-program the same chip using electrical signals
- ❑ The information contained in the EEPROM memory can be altered by using electrical signals at the register/Byte level
- ❑ They can be erased and reprogrammed within the 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

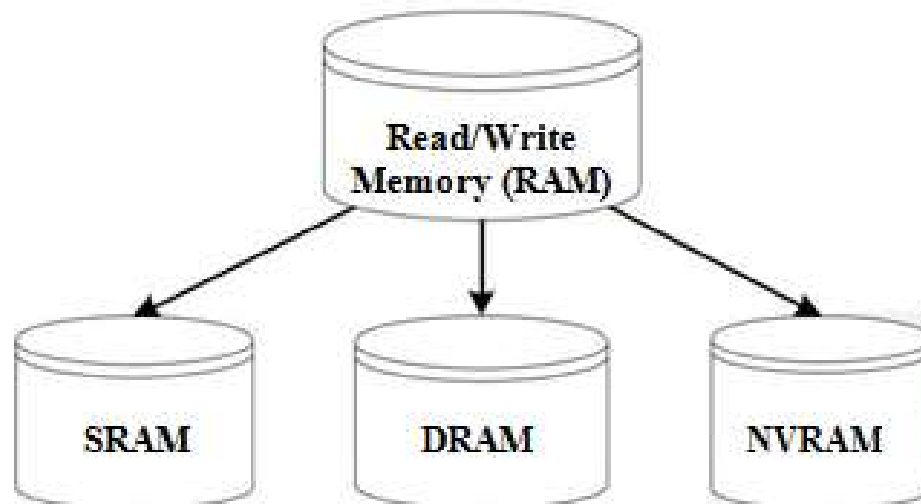
- ❑ FLASH memory is a variation of EEPROM technology.
- ❑ FLASH is the latest ROM technology and is the most popular ROM technology used in today's embedded designs
- ❑ It combines the re-programmability of EEPROM and the high capacity of standard ROMs
- ❑ FLASH memory is organized as sectors (blocks) or pages
- ❑ FLASH memory 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/page should be erased before re-programming

READ-WRITE MEMORY/RANDOM ACCESS MEMORY(RAM)

- ❑ RAM is the data memory or working memory of the controller/processor
- ❑ RAM is a **volatile memory** , which means the data is stored as long as the power is on, when the power is turned off, all the contents are destroyed.
- ❑ RAM is a **Direct Access Memory**, which means the memory location can access the directly without traversing through the entire memory locations. (i.e. Random Access of memory location).

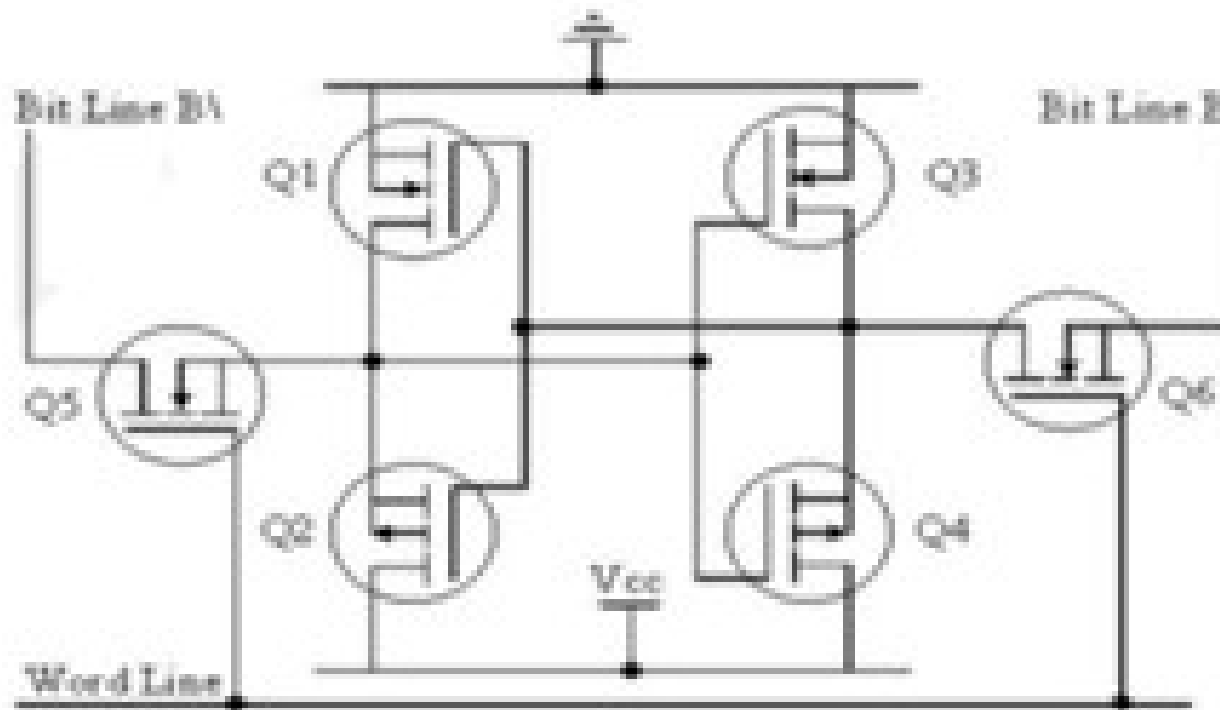
RAM are generally of 3 categories

1. SRAM (Static Random Access Memory)
2. DRAM (Dynamic Random Access Memory)
3. NVRAM (Non-Volatile RAM)



-
- ❑ SRAM Stores data in the form of voltage.
 - ❑ SRAM are made up of Flip-Flops.
 - ❑ SRAM is the fastest form of RAM
 - ❑ SRAM is realized using 6 transistors (MOSFET),
 - ❑ where 4 transistors are used for latch(flip-flop) and 2 to control the Access.

SRAM can be visualized as follow:



SRAM cell implementation

DYNAMIC RANDOM ACCESS MEMORY (DRAM)

- ❑ Dynamic RAM stores data in the form of charge.
- ❑ They are made up of MOS transistor gates.

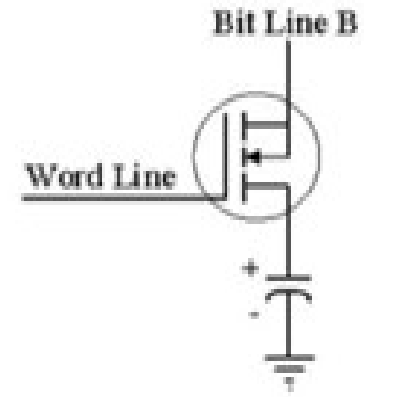
Advantages of DRAM are:

- ❖ high density
- ❖ low cost

Disadvantage of DRAM are:

- ❖ Information is stored as charge it gets leaked off with time and
- ❖ to prevent this they need to be refreshed periodically

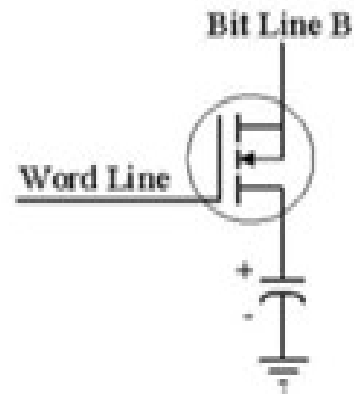
- ❑ Special circuits called DRAM controllers are used for the refreshing operation. The refresh operation is done periodically in milliseconds interval.



DRAM cell implementation

DRAM CELL IMPLEMENTATION

- ❑ MOSFET acts as the gate for incoming and outgoing data.
- ❑ Capacitor acts as a bit storage.



DRAM cell implementation

DIFFERENCE BETWEEN SRAM AND DRAM

SRAM Vs 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)
More expensive	Less Expensive
Fast in operation. Typical access time is 10ns	Slow in operation due to refresh requirements. Typical access time is 60ns. Write operation is faster than read operation.

NVRAM (NON-VOLATILE RAM)

- ❑ NVRAM is a Random Access Memory with Battery Backup.
- ❑ NVRAM contains the Static RAM and a minute battery for providing supply to the memory in the absence of the Power Supply.
- ❑ Memory and Battery are packed together in a single chip.

MEMORY SELECTION FOR EMBEDDED SYSTEMS

- ❑ Selection of suitable memory is very much essential step in high performance applications, since the challenges and limitations of the system performance are often decided upon the type of memory architecture.
- ❑ Systems memory requirement depend primarily on the nature of the application, Memory performance and capacity requirement for low cost systems.

FACTORS CONSIDERED WHILE SELECTING THE MEMORY

❑ Factors considered while selecting the memory devices are:

- ❖ Speed
- ❖ Data storage size and capacity
- ❖ Bus width
- ❖ Power consumption
- ❖ Cost

EMBEDDED SYSTEM REQUIREMENTS

- ❑ *Program memory* for holding control algorithm or embedded OS and the applications designed to run on top of OS.
- ❑ *Data memory* for holding variables and temporary data during task execution.
- ❑ *Memory for holding non-volatile data* which are modifiable by the application.

-
- ❑ Embedded system is designed using SOC or microcontroller (with on chip RAM &ROM) , then the size of on-chip memory depends on the type of the Application used.
 - ❑ On-chip memory is not sufficient then how much external memory need to be interfaced.

-
- ❑ ES design is RTOS based ,the RTOS requires certain amount of RAM for its execution and ROM for storing RTOS Image.

 - ❑ A smart phone device with windows OS is typical example for Embedded device requires say 512MB RAM and 1GB ROM are minimum requirements for running the mobile device.

❑ Memory chips are available in standard sizes like 512 bytes,1KB,2KB ,4KB,8KB,16 KB1MB etc.

❑ **Flash Memory** is the Ideal choice for ROM in Embedded Systems.

❑ Flash memory available in two major variants

❖ **NAND FLASH**

❖ **NOR FLASH**

-
- ❑ **NAND FLASH** is a high density low cost non-volatile storage memory.
 - ❑ **NOR FLASH** is less dense and slightly expensive but supports Execute in place(XIP).
 - ❑ The XIP technology allows the execution of code memory from ROM itself, without copying it to the RAM.
 - ❑ **NAND Flash** doesn't support XIP, if used to store program code, **DRAM** is used for executing the code.
 - ❑ Better performance is achieved by combining NOR and NAND Flash.

SENSORS AND ACTUATORS

- ❑ Embedded System are used in the **real time world** for Controlling/monitoring functions which is achieved in accordance with the changes happening to the **Real World**.
- ❑ The changes in the system environment or variables are detected by the sensors connected to the input port of the embedded system.
- ❑ The changes in the control variable is achieved through an actuator connected to the out port of the embedded system.

SENSORS:

Sensors:

- A transducer device which converts energy from one form to another for any measurement or control purpose.
- Sensors acts as input device

Ex: Sensors used to count the number of steps



ACTUATORS:

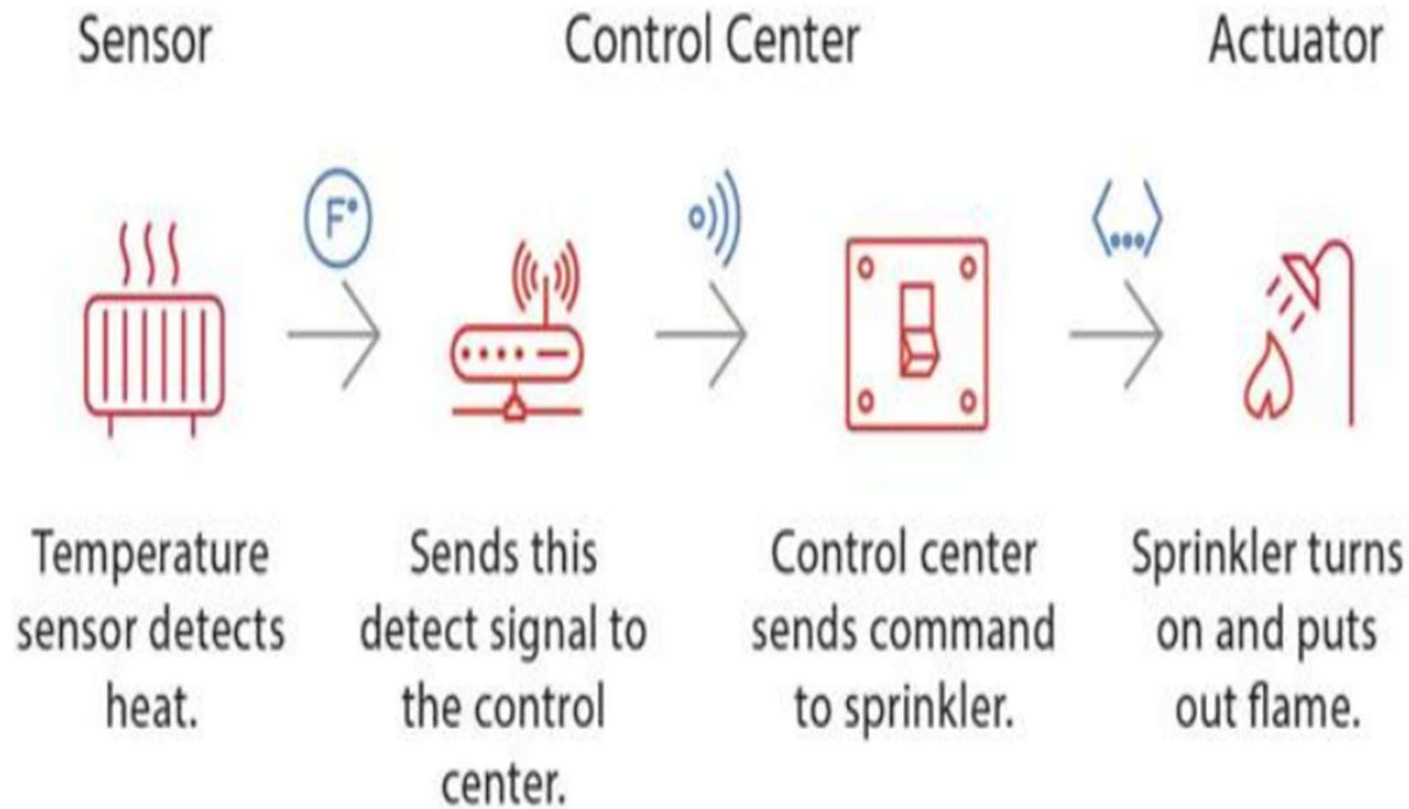
Actuators:

- A form of transducer device (mechanical or electrical) which converts signals to corresponding physical action (motion).
- ❑ Actuator acts as an output device
 - ❖ Ex: Checking the surrounding light intensity and increase or decrease the intensity of the light.



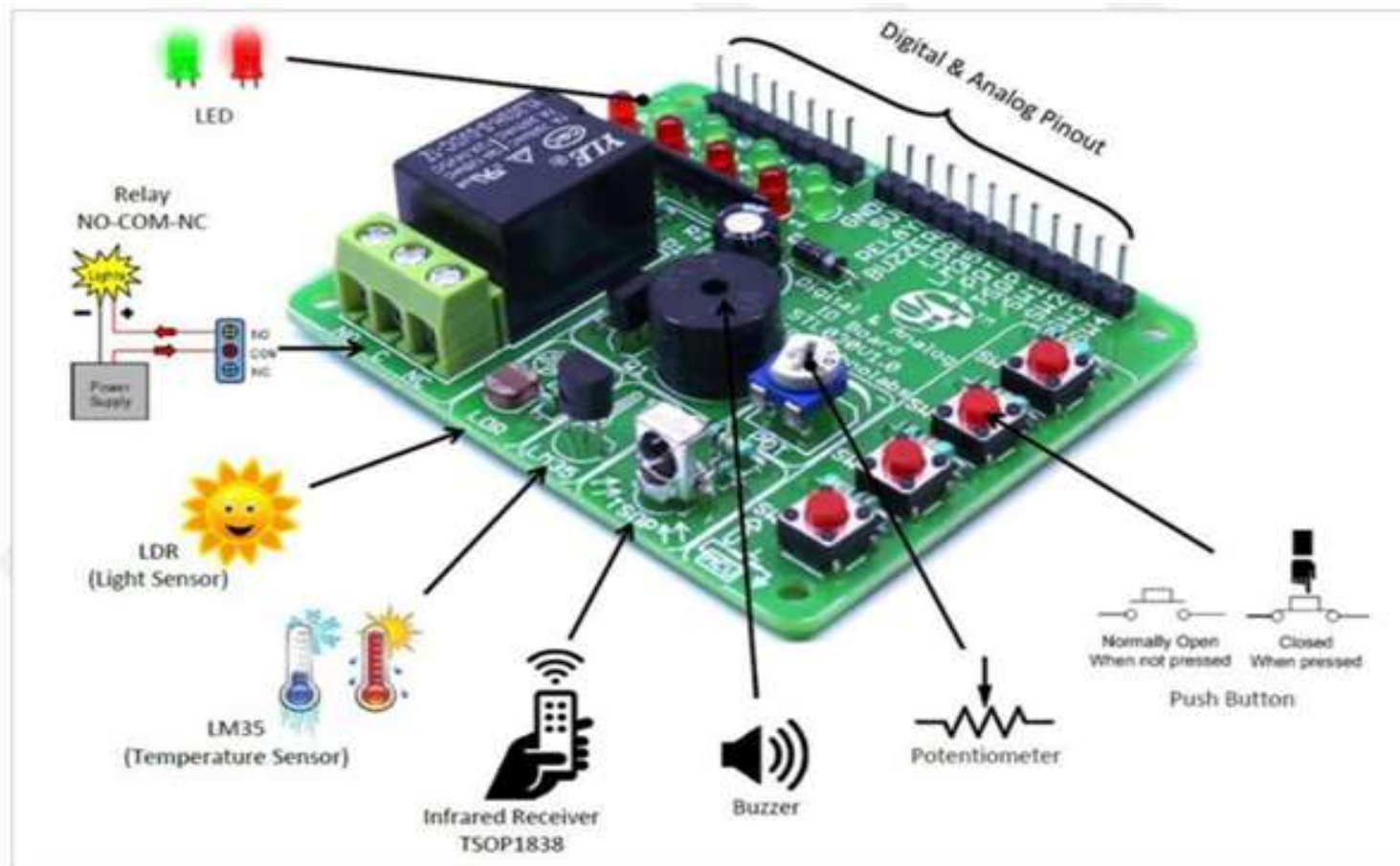
SENSOR TO ACTUATOR FLOW

□ The following Figure shows the sensor to actuator flow:



DIFFERENCES B/W SENSORS & ACTUATORS

Sensors	Actuators
1. Sensor is an input device	1. Actuator is an output device
2. Convert a physical parameter to an electrical output	2. Convert an electrical signal to a physical output
3. A device that detects events or changes in the environment and send the information to another electronic device	3. A component of a machine that is responsible for moving and controlling mechanisms
4. Sensor help to monitor the changes in the environment	4. Actuator helps to control the environment or physical changes



Silicon TechnoLabs Digital Analog Arduino Starter kit

THE I/O SUBSYSTEMS

- ❑ Facilitates the interaction with the outside world.
- ❑ Interaction will happen using Sensors and Actuators connected to the input and output port.
- ❑ The Sensors are not directly connected to the input port, but they are interfaced using translating systems like ADC, optocouplers, etc.

□ The I/O systems which we are going to discuss are

1. LED
2. 7 Segment LED Display
3. Stepper motors
4. Push Button Switch
5. Keyboard
6. Programmable Peripheral Interface (PPI)

LIGHT EMITTING DIODE (LED):

Light Emitting Diode (LED):

- ❑ Used as a visual indicator in most of the Embedded System.
- ❑ Used for indication for a status of device. “Device On” or “Device off”
- ❑ A *light-emitting diode (LED)* is a semiconductor light source that emits light when current flows through it.

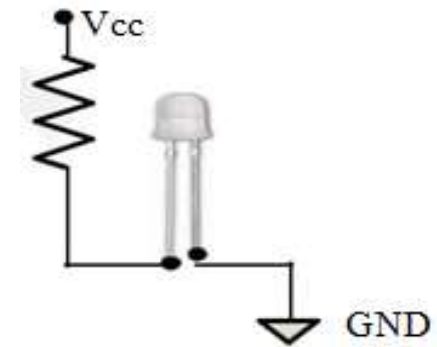
❑ For proper functioning of the LED, the Anode should be connected to +ve terminal and cathode should be connected to -ve terminal.



LED INTERFACING

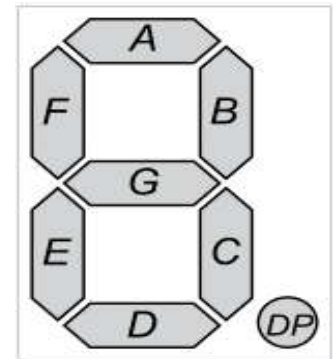
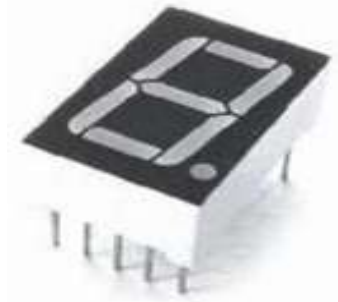
LED is interfaced with processor/controller in 2 ways.

1. Source mode: The Anode is directly connected to the port pin and the port pins drives the LED.(port pins sources the current for logic high).
2. Sink mode: Cathode of LED is connected to the port pin of the processor/ controller and anode to the supply voltage through the limiting resistor,(LED is turned on when the port pin at logic low)



7 SEGMENT LED DISPLAY:

- ❑ It is an output device for displaying alpha numeric characters .
- ❑ It contains 8 segments of which 7 segment (A-G) is used to display alpha numeric character and 1 segment (DP) is used to represent “Decimal point”.
- ❑ To display numbers and characters, LED segments A to G and DP must be lit accordingly. Example:
 - ❖ 4 → Segments F, G, B, C
 - ❖ 3 → Segments A, B, C, D, G, DP
 - ❖ d → Segments B, C, D, E, G
- ❑ The 7 Segment display are of 2 types
 - ❖ *Common Anode type*
 - ❖ *Common Cathode type*



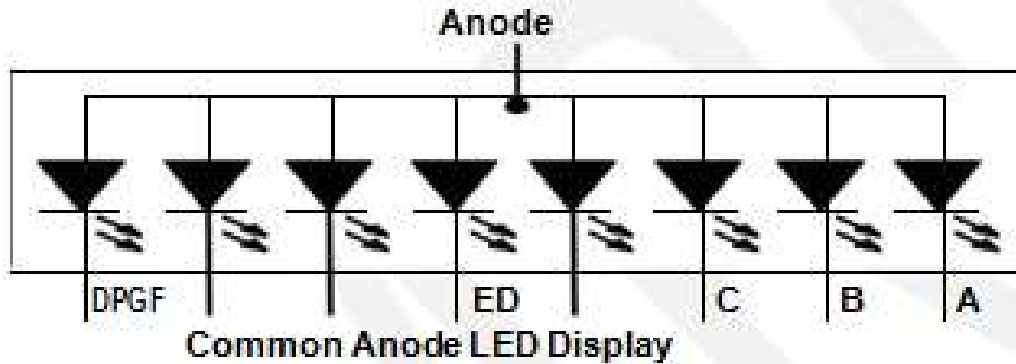
Connections:

- Each LED segment (A to G & DP) is connected to a processor/controller port.
- Segment 'A' connects to the least significant port pin (smallest value).
- Segment 'DP' connects to the most significant port pin (largest value).

Current and Resistors:

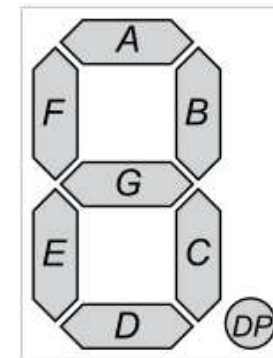
- Each segment typically uses 20mA of current.
- To prevent excessive current flow, current-limiting resistors are connected to the anode/cathode of each segment.
- Resistor values are calculated based on the LED display's electrical parameters.

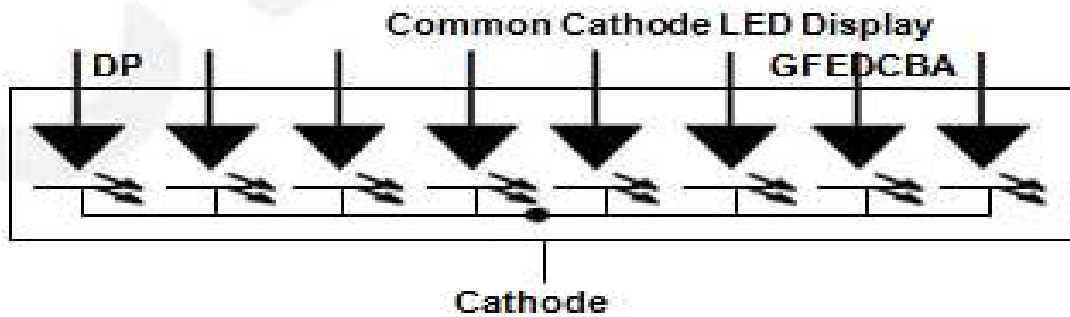
COMMON ANODE LED DISPLAY



- ❑ In a **Common Anode** 7-segment LED display, the **anode** (positive side) of all segments is connected to a **shared 5V supply** through a current-limiting resistor.
- ❑ Each **segment's cathode** (negative side) is connected to the respective **port pin** of the processor/controller.

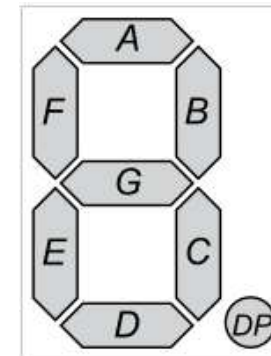
Character	G	F	E	D	C	B	A	Hex.C ode
0								
1	0	0	0	0	0	1	1	06H
2								
3	0	1	0	0	1	1	1	6FH
A								





Character		A	B	C	D	E	F	G	Hex.Code
0	1								
1	1								
2	1								
3	1	1	1	1	1	0	0	1	F9H
A	1								

- ❑ In a Common Cathode 7-segment LED display, the cathode (negative side) of all segments is connected together and linked to ground (0V).
- ❑ Each segment's anode (positive side) is connected to a respective port pin of the processor/controller.
- ❑ To turn on a segment, the port pin connected to its anode must be set to logic 1 (HIGH).



OPTOCOUPPLERS:

An **optocoupler** is an electronic component used to **isolate** two parts of a circuit while allowing signals to pass between them.

How It Works:

- ❑ It consists of an **LED** (light source) and a **phototransistor** (light receiver) inside a single package.
- ❑ The LED lights up when a signal is sent.
- ❑ The light activates the phototransistor, allowing the signal to continue—but without direct electrical connection.

Why Is It Useful?

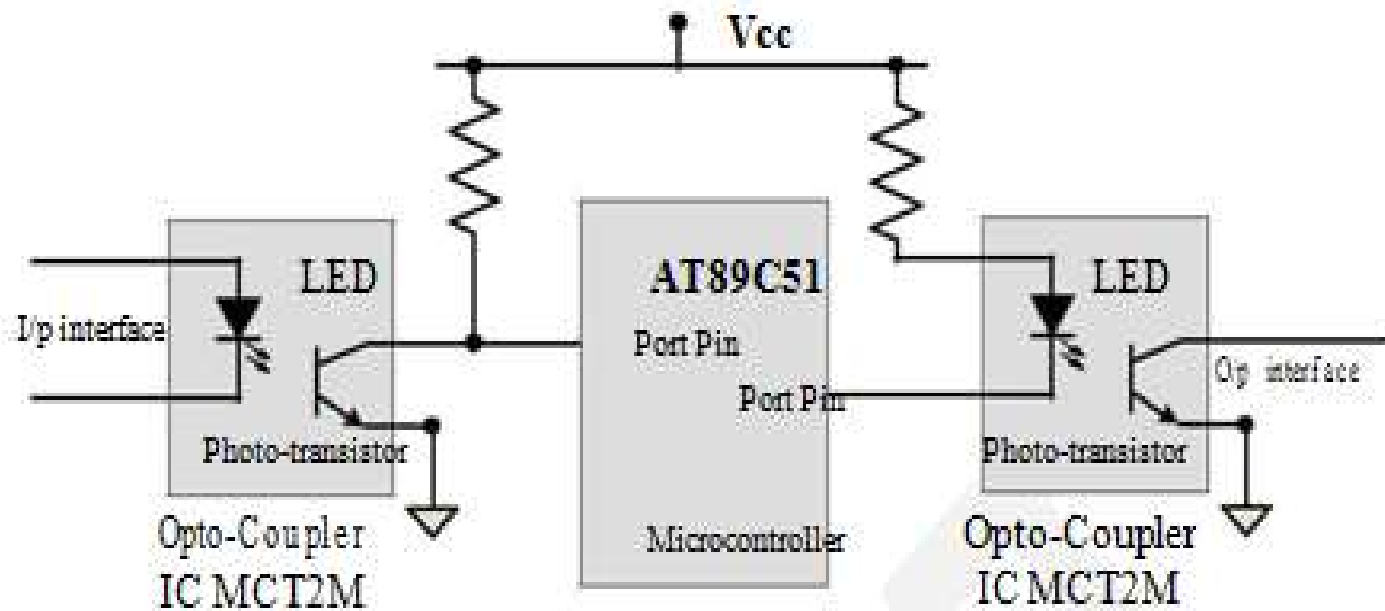
Optocouplers help in:

- ❖ **Preventing interference** in data communication.
- ❖ **Separating high and low voltage circuits** for safety.
- ❖ **Isolating circuits** to avoid electrical damage.
- ❖ **Boosting signal strength** in some applications.

OPTOCOUPPLERS:

Optocouplers:

- It is a circuit which isolate 2 part of the circuit.
- It combines LED and phototransistor in a single housing package.
- It is
 - used to suppress interference in Data Communication,
 - high voltage separation,
 - circuit isolation and
 - signal intensification.
- They are used either at the input circuit or at the output circuit.



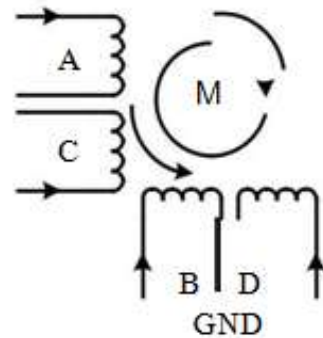
Optocoupler in input and output circuit

STEPPER MOTOR:

- ❑ A stepper motor is an Electro-mechanical device which produces a discrete rotation when an input voltage applied to it.
- ❑ Stepper motors are widely used in Industrial Embedded System, Consumer Electronics and Robotics control system.
- ❑ Based on the *winding* used Stepper motor are classified as
 - ❑ *Unipolar*
 - ❑ *Bipolar*

UNIPOLAR

- ❑ A Unipolar stepper motor contains two windings per phase.
- ❑ The direction of rotation (clockwise or anticlockwise) of a stepper motor is controlled by changing the direction of current flow.
- ❑ Current in one direction flows through one coil and current flows in the opposite direction through the other coil.
- ❑ It is easy to shift the direction of rotation by just switching the terminals to which the coils are connected



- ❑ The coils are represented as A, B, C and D.
- ❑ Coils A and C carry current in opposite directions for phase 1 (only one of them will be carrying current at a time).
- ❑ Similarly, B and D carry current in opposite directions for phase 2 (only one of them will be carrying current at a time)

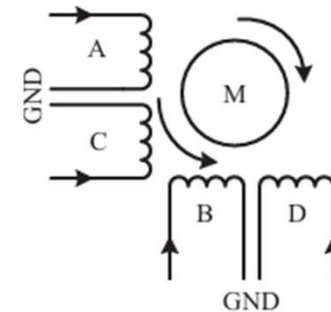


Fig. 2.18 2-Phase unipolar stepper motor

BIPOLAR

- ❑ A bipolar stepper motor contains single winding per phase.
- ❑ For reversing the motor rotation the current flow through the windings is reversed dynamically.
- ❑ It requires complex circuitry for current flow reversal

- ❑ The stator winding details for a two phase unipolar stepper motor is shown in
- ❑ The stepping of stepper motor can be implemented in different ways by changing the sequence of activation of the stator windings.
- ❑ The different stepping modes supported by stepper motor are explained below.
 - ❑ 1. *Full Step*
 - ❑ 2. *Wave step*
 - ❑ 3. *Half step*

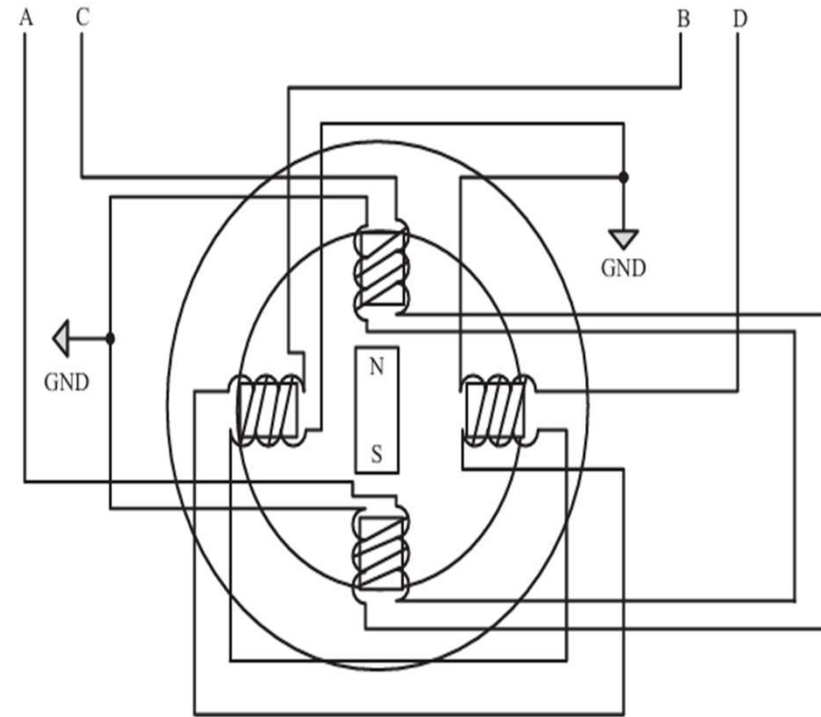


Fig. 2.19 Stator Winding details for a 2 Phase unipolar stepper motor

Full Step In the full step mode both the phases are energised simultaneously. The coils A, B, C and D are energised in the following order:

Step	Coil A	Coil B	Coil C	Coil D
1	H	H	L	L
2	L	H	H	L
3	L	L	H	H
4	H	L	L	H

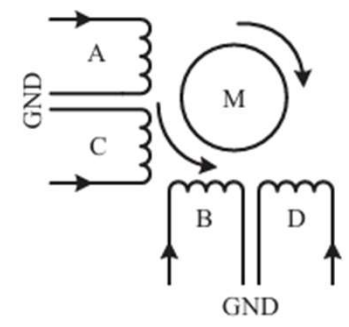


Fig. 2.18 2-Phase unipolar stepper motor

It should be noted that out of the two windings, only one winding of a phase is energised at a time.

□ Wave step:

□ In the wave step mode only one phase is energised at a time and each coils of the phase is energised alternatively. The coils A, B, C, and D are energised in the following order:

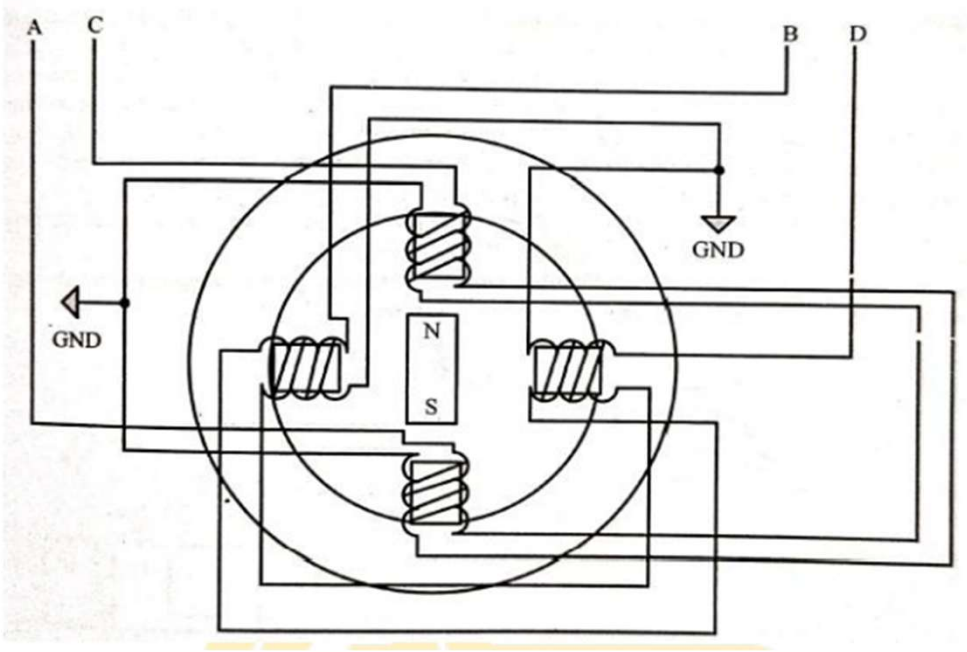
Step	Coil A	Coil B	Coil C	Coil D
1	H	L	L	L
2	L	H	L	L
3	L	L	H	L
4	L	L	L	H

-
- ❑ **Half Step:** It uses the combination of wave and full step.
 - ❑ It has the highest torque and stability.
 - ❑ The coil energizing sequence for half step is given below.

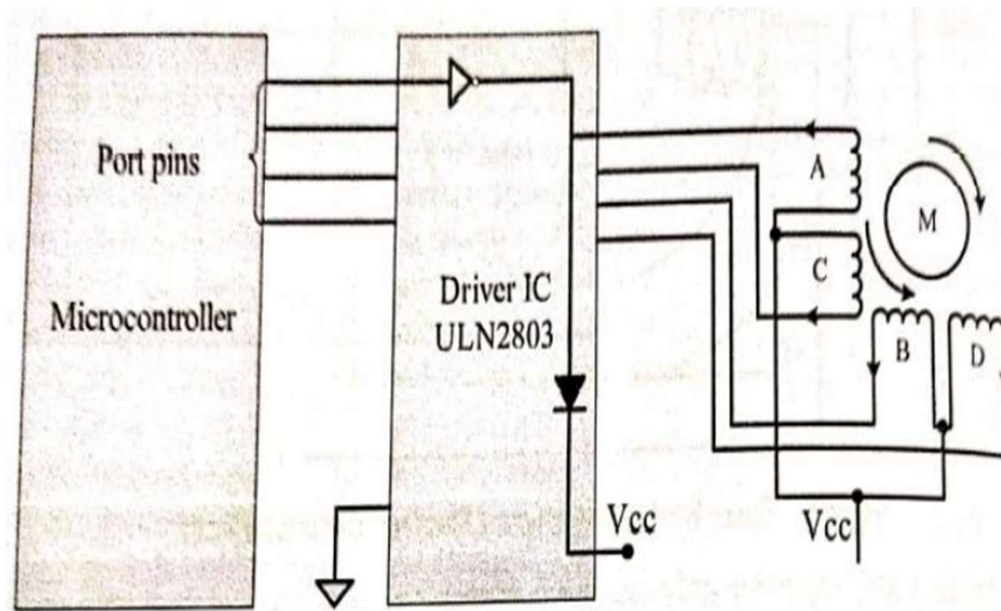
Step	Coil A	Coil B	Coil C	Coil D
1	H	L	L	L
2	H	H	L	L
3	L	H	L	L
4	L	H	H	L
5	L	L	H	L
6	L	L	H	H
7	L	L	L	H
8	H	L	L	H

❑ The rotation of the stepper motor can be reversed by reversing the order in which the coil is energized.

❑ The following Figure shows the stator winding details of Stepper motor:

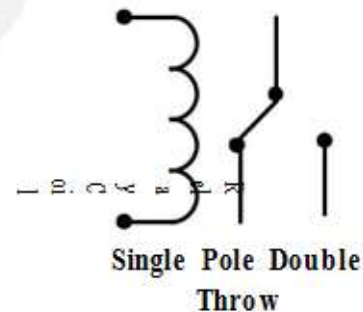
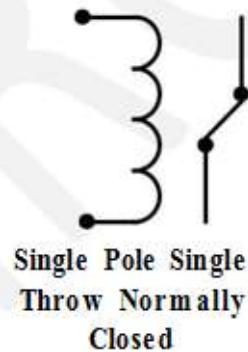
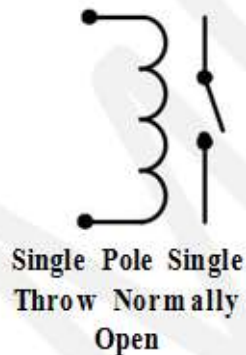


-
- ❑ Two-phase unipolar stepper motors are the popular choice for embedded applications.
 - ❑ Also the supply voltage required to operate stepper motor varies normally in the range 5V to 24V.
 - ❑ Depending on the current and voltage requirements, special driving circuits are required to interface the stepper motor with microcontroller/ processors.
 - ❑ The following circuit diagram illustrates the interfacing of a stepper motor through a driver circuit connected to the port pins of a microcontroller/ processor



RELAY

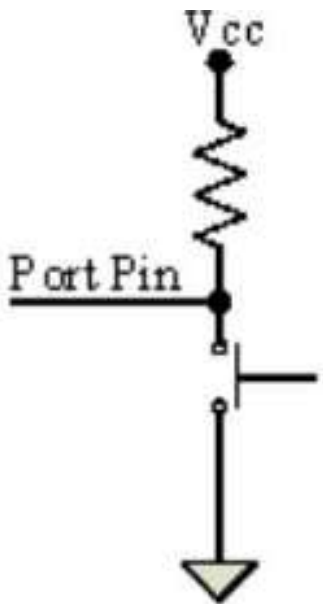
- ❑ Relay is an Electro-mechanical device.
- ❑ It Act as a dynamic path selectors for signals and power.
- ❑ Relay works on Electro-magnetic Principle.
- ❑ When Voltage is applied to relay coil, current flows through the coil which in turn generates the magnetic field, which attracts the armature core and moves the contact point.



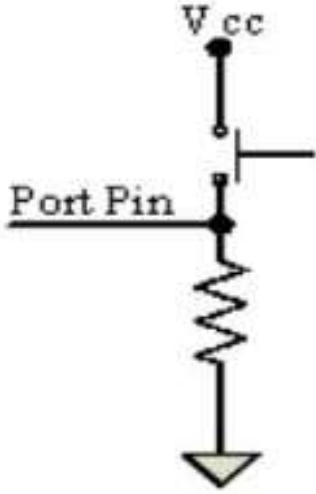
Push Button Switch

- ❑ It acts as an Input Device
- ❑ Push Button Switch comes in 2 configurations
 1. *Push to make*
 2. *Push to break*
- ❑ ***Push to Make***: Normally open, make a contact when switch is pressed.
- ❑ ***Push to Break***: Normally closed, it break the contact when switch is pressed.
- ❑ Push button is normally used for generating a monetary pulse.

❑ Push Button can generate either low pulse or high pulse depending on the circuitry as shown.



'LOW' Pulse Generator

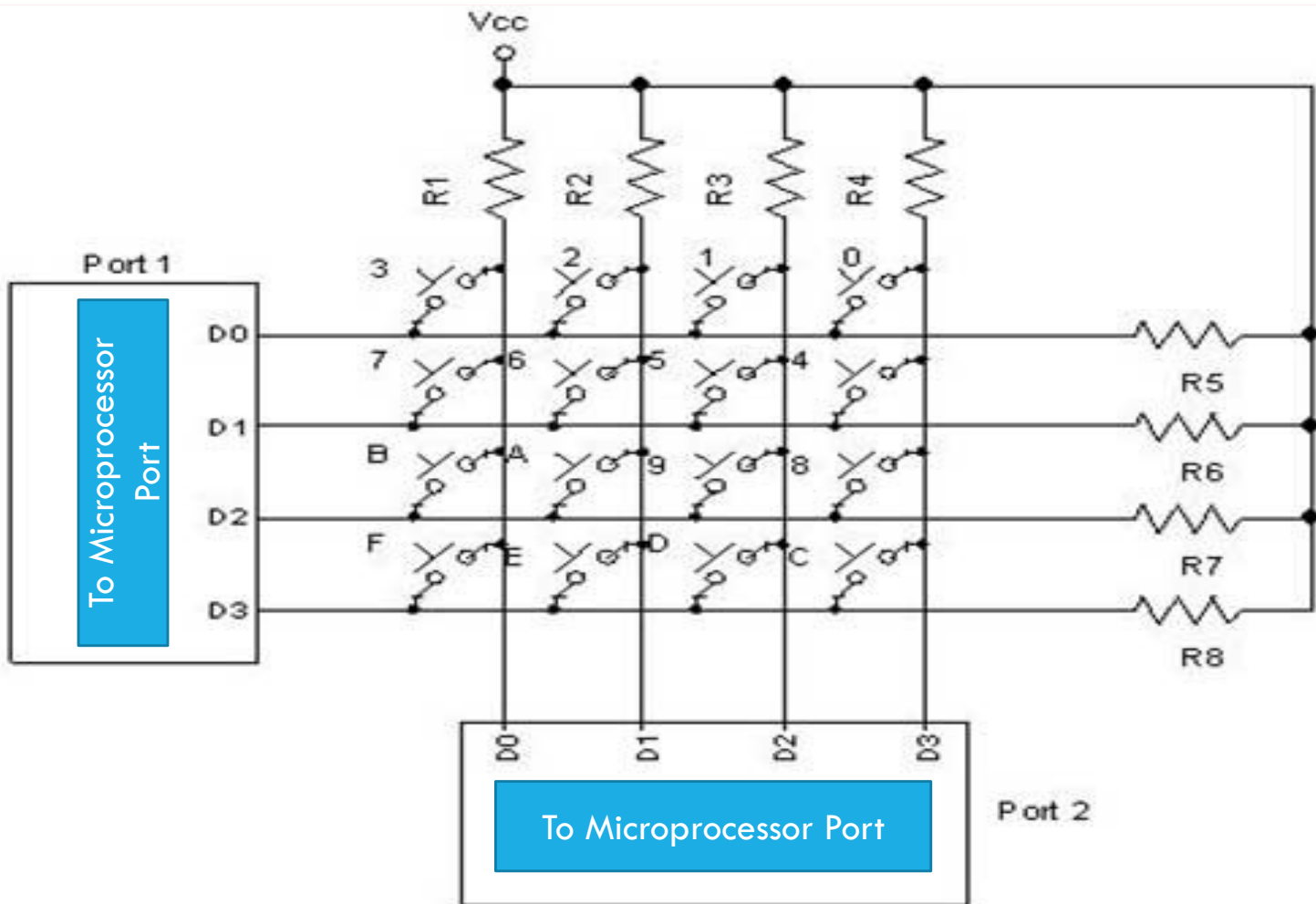


'HIGH' Pulse Generator



Keyboard

- ❑ When keys required are more, push button is not possible.
- ❑ Keyboard is the input device for user interfacing.
- ❑ Used when an Embedded System requires large number of input pins.
- ❑ The Pins are arranged in the matrix form
- ❑ 4 Rows and 4 Columns, which can afford upto 16 pins.
- ❑ Keyboard uses scanning techniques, by pulling row matrix to low, it will read column.
- ❑ After reading each column, column is hold low and row is held high to read the next row.



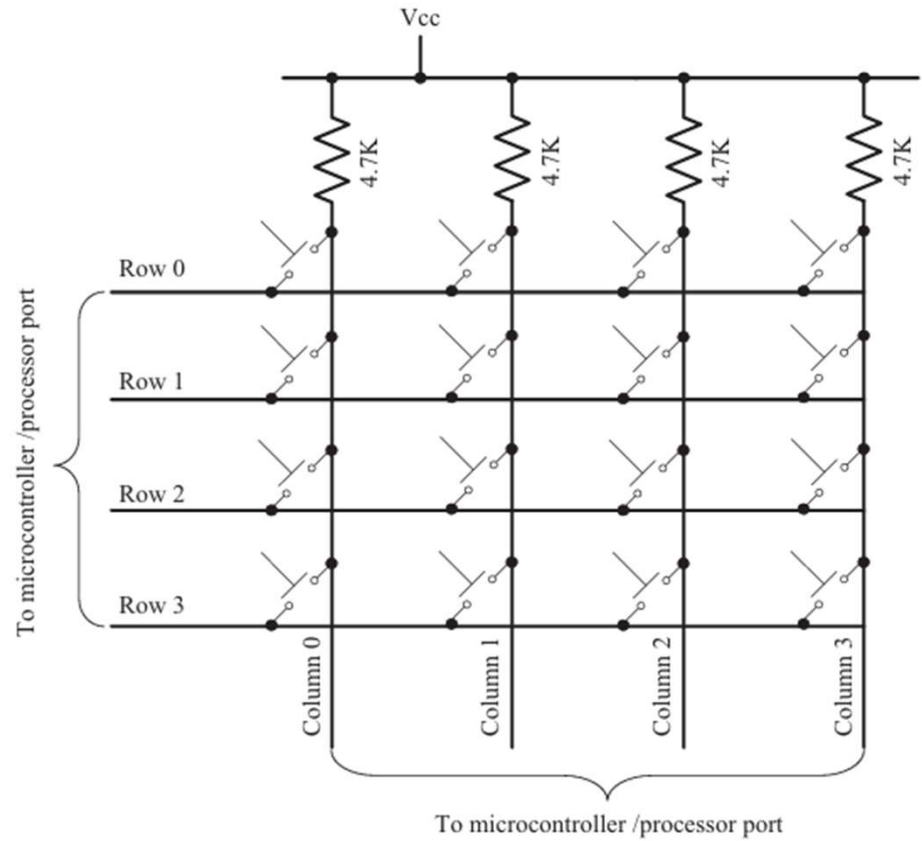


Fig. 2.24 Matrix keyboard Interfacing

-
- ❑ The Process is repeating till the key pressed is held at low.
 - ❑ The keys used are mechanical device, so the debounce issues will occur..
 - ❑ To Overcome the debouncing, appropriate debouncing circuit will be applied

COMMUNICATION INTERFACE

- ❑ Communication interface is essential for communicating with various subsystems of the embedded system and with the external world.
- ❑ The communication interface can be viewed in two different perspectives.
 1. Device/board level communication interface (Onboard Communication Interface)
 2. Product level communication interface (External Communication Interface)

DEVICE/BOARD LEVEL COMMUNICATION INTERFACE (ONBOARD COMMUNICATION INTERFACE)

- ❑ The communication channel which interconnects the various components within an embedded product is referred as Device/board level communication interface (*Onboard Communication Interface*)
- ❑ Examples: Serial interfaces like I2C, SPI, UART, 1-Wire etc and Parallel bus interface

PRODUCT LEVEL COMMUNICATION INTERFACE (EXTERNAL COMMUNICATION INTERFACE)

- ❑ The “*Product level communication Interface*” (*External Comm. Interface*) is responsible for data transfer between the embedded system and other devices or modules.
- ❑ The external communication interface can be either wired media or wireless media and it can be a serial or parallel interface.
- ❑ Examples for *wireless communication interface*:
 - ❖ Infrared (IR), Bluetooth (BT), Wireless LAN (Wi-Fi), Radio Frequency waves (RF), GPRS etc.
- ❑ Examples for *wired interfaces*:
 - ❖ RS-232C/RS-422/RS-485, USB, Ethernet (TCP-IP), IEEE 1394 port, Parallel port etc.

ONBOARD COMMUNICATION INTERFACES

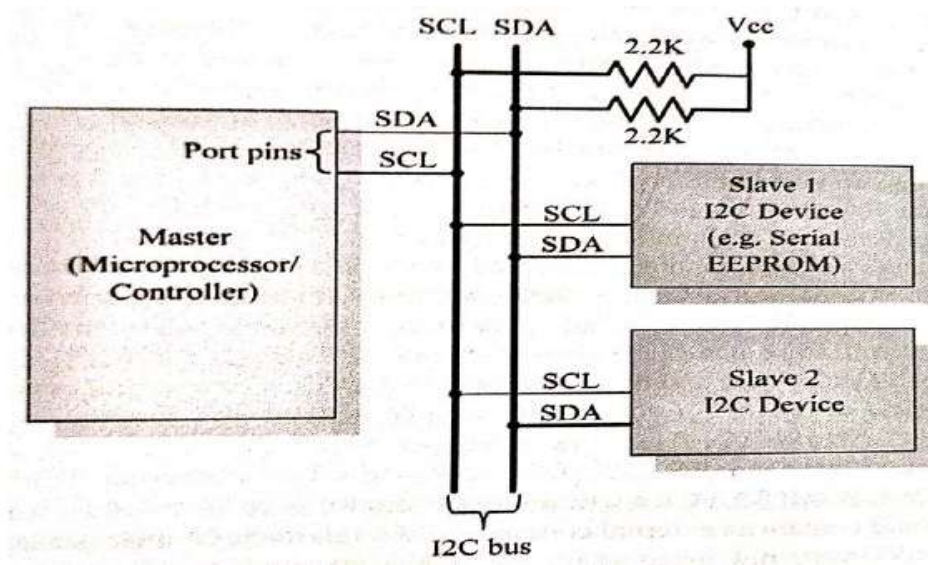
- ❑ Onboard Communication Interface refers to the different communication channels/buses for interconnecting the various integrated circuits and other peripherals within the embedded system.
- ❑ Inter Integrated Circuit(I²C) Interfaces
- ❑ Serial Peripheral Interface(SPI) Bus
- ❑ Universal Asynchronous Receiver Transmitter (UART)
- ❑ 1-Wire Interface
- ❑ Parallel Interface

INTER INTEGRATED CIRCUIT BUS(I2C/I2C))

- ❑ **Inter Integrated Circuit Bus** (I²C – Pronounced, I square C) is a synchronous bi-directional half duplex two wire serial interface bus.
- ❑ The concept of I²C bus was developed by “**Philips Semiconductors**” in the early 1980“s. The original intention of I²C was to provide an easy way of connection between a microprocessor/microcontroller system and the peripheral chips in Television sets.
- ❑ The I²C bus is comprised of two bus lines, namely:
 - ❖ Serial Clock – SCL
 - ❖ Serial Data – SDA

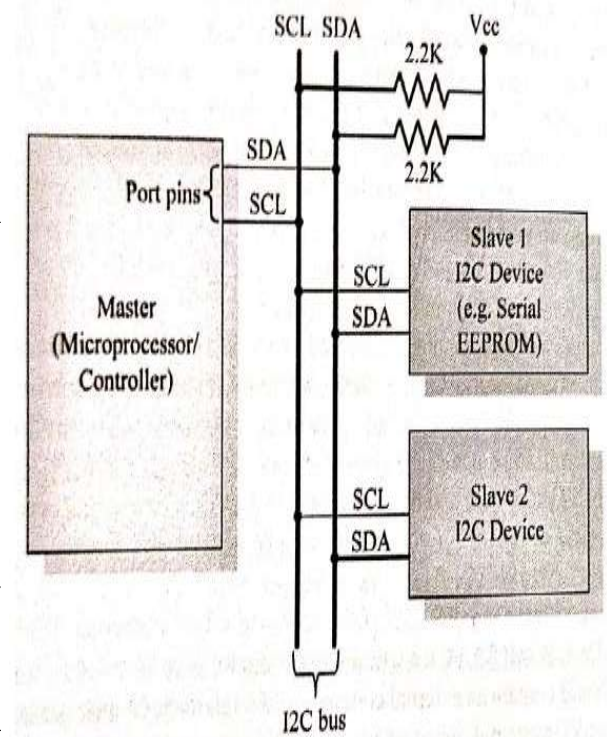
-
- ❑ I²C bus is a shared bus system to which many number of I²C devices can be connected.
 - ❑ Devices connected to the I²C bus can act as either 'Master' device or 'Slave' device.
 - ❑ The 'Master' device is responsible for
 - ❖ controlling the communication by initiating/terminating data transfer,
 - ❖ sending data and
 - ❖ generating necessary synchronization clock pulses.
 - ❑ The 'Slave' device wait for the commands from the master and respond upon receiving the commands.

- ❑ 'Master' and 'Slave' devices can act as either transmitter or receiver, regardless of whether a master is acting as transmitter or receiver, the synchronization clock signal is generated by the 'Master' device only.
- ❑ I²C supports multi-masters on the same bus.

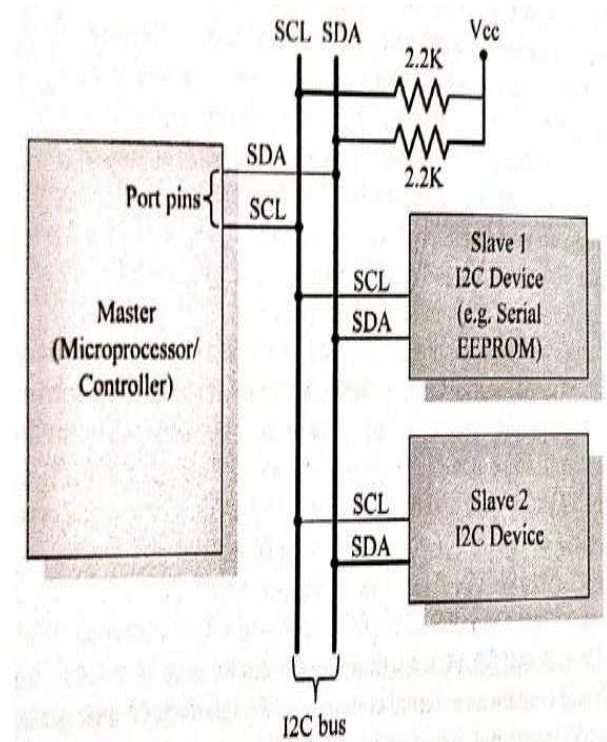


THE SEQUENCE OF OPERATIONS FOR COMMUNICATING WITH AN I2C SLAVE :

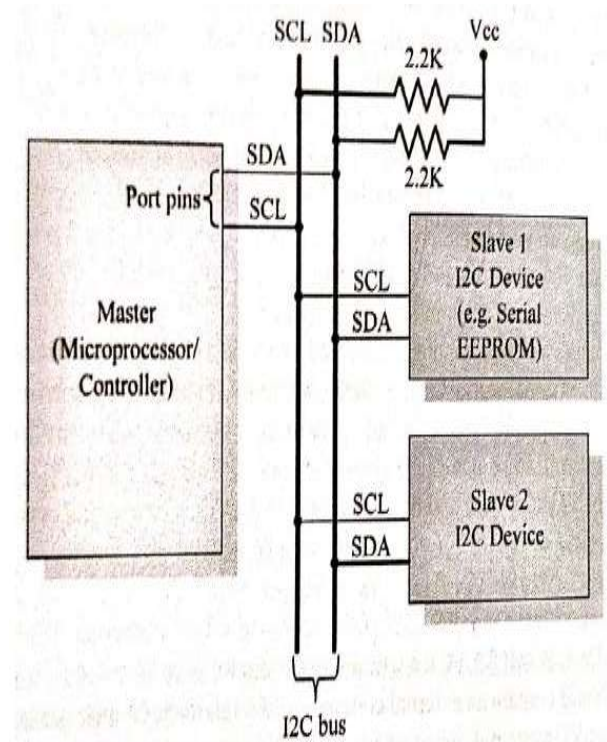
1. The Master device pulls the clock line (SCL) of the bus to “**HIGH**”.
2. The Master Device pulls the data line (SDA) “**LOW**”. When the SCL line is at logic “**HIGH**”(Start Condition for data transfer)
3. The Master device sends the address (7 – 10 bit wide) of the “**Slave**” device to which it wants to communicate, over the SDA line. Clock pulses are generated at the SCL line for synchronizing the bit reception by the slave device. The MSB of the data is always transmitted first. The data in the bus is valid during the “**HIGH**” period of the clk signal.



4. The **Master device** sends the **Read** or **Write** Bit (Bit = 1, Read operation; Bit = 0, Write Operation) according to the requirement.
5. The **Master device** waits for the **Acknowledge** bit from the slave device.
6. The **Slave device** with the address requested by the **Master device** responds by sending the **Acknowledge** bit (Bit = 1) over the **SDA** line.
7. Upon receiving the **Acknowledge** bit, the **Master device** send the 8 bit data to the **Slave** device over **SDA** line, if the requested operation is “**Write to Device**” If the requested operation is “**Read from device**”, the slave device sends data to the master over the SDA line.



8. The **Master** device waits for the **Acknowledgement** bit from the device upon byte transfer complete for a write operation and sends an **Acknowledge** bit to the Slave device for the Read operation.
9. The **Master device** terminates the transfer by pulling the **SDA** line “**HIGH**” when the clock line **SCL** is at logic “**HIGH**” (Indicating the stop condition).

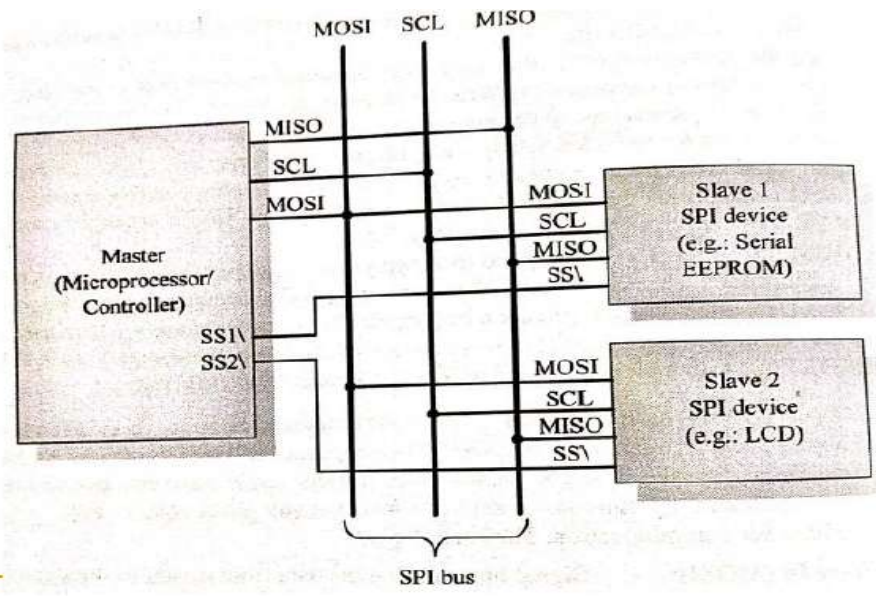


Serial Peripheral Interface Bus(SPI)

- ❑ SPI is a synchronous bi-directional full duplex 4 wire serial interface bus.
- ❑ The Concept was first introduced by Motorola.
- ❑ SPI is a single master multi slave systems.
- ❑ It can have multi master, but only one master is active at any given point of time.

- ❑ SPI gives 4 signal lines for communication. They are
- ❑ **Master out slave in (MOSI)** : Signal lines carrying the data from master to slave device.
- ❑ **Master in Slave Out (MISO)**: Signal lines carrying the data from slave to master device.
- ❑ **Serial Clock (SCLK)**: Signal lines carrying clock signals
- ❑ **Slave Select(SS)**: Signal line for Slave device select. It is an Active low signal.

The bus interface fig illustrating the connection of Master and slave devices on the SPI bus.



WORKING

- ❑ The Master device is responsible for generating the clock Signals. It Select's the required slave device by placing "low " on Slave's Select Signal.
- ❑ SPI works on the principles of Shift Registers.
- ❑ The master and slave devices contain a special shift register for the data to transmit or receive.
- ❑ The size of the shift register is device dependent.
- ❑ Normally it is a multiple of 8.
- ❑ During transmission from the master to slave , the data in the master's shift register is shifted out to the MOSI pin and it enters the shift register of the slave device through the MOSI pin of the slave device.

-
- ❑ At the same time, the shifted out data bit from the slave device's shift register enters the shift register of the master device through MISO pin.
 - ❑ In summary ,the shift registers of “Master" and "Slave" devices form a circular buffer.
 - ❑ When compared to I2C, SPI bus is most suitable for applications requiring transfer of data in “Streams”.
 - ❑ The only limitation is SPI doesn't support an Acknowledgement mechanism.

Difference between I2C and SPI

I2C	SPI
Speed limit varies from 100kbps, 400kbps, 1mbps, 3.4mbps depending on i2c version.	More than 1mbps, 10mbps till 100mbps can be achieved.
Half duplex synchronous protocol	Full Duplex synchronous protocol
Support Multi master configuration	Multi master configuration is not possible
Acknowledgement at each transfer	No Acknowledgement
Require Two Pins only SDA, SCL	Require separate MISO, MOSI, CLK & CS signal for each slave.
Addition of new device on the bus is easy	Addition of new device on the bus is not much easy a I2C
More Overhead (due to acknowledgement, start, stop)	Less Overhead
Noise sensitivity is high	Less noise sensitivity

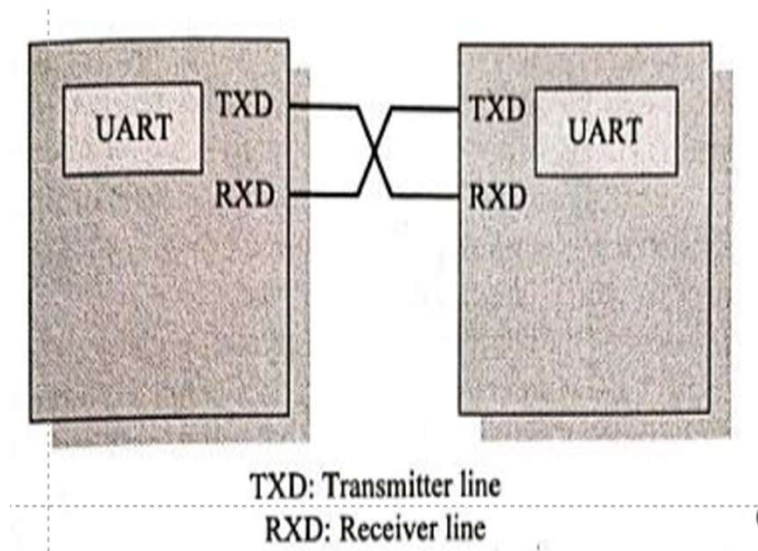
Universal Asynchronous Receiver Transmitter(UART)

- ❑ **Universal Asynchronous Receiver Transmitter (UART):** *Universal Asynchronous Receiver Transmitter (UART)* based data transmission is an **asynchronous** form of serial data transmission.
- ❑ UART based serial data transmission ***doesn't require a clock signal to synchronize*** the transmitting end and receiving end for transmission.
- ❑ Instead it relies upon the pre-defined agreement between the transmitting device and receiving device.
- ❑ ***The serial communication settings*** (Baud rate, number of bits per byte, parity, number of start bits and stop bit and flow control) for both transmitter and receiver should be set as identical.
- ❑ The start and stop of communication is indicated through inserting special bits in the data stream.

-
- ❑ While sending a byte of data, a start bit is added first and a stop bit is added at the end of the bit stream.
 - ❑ The least significant bit of the data byte follows the 'start' bit.
 - ❑ The 'start' bit informs the receiver that a data byte is about to arrive.
 - ❑ The receiver device starts polling its 'receive line' as per the baud rate settings.
 - ❑ If the baud rate is 'x' bits per second, the time slot available for one bit is $1/x$ seconds.
 - ❑ The receiver unit polls the receiver line at exactly half of the time slot available for the bit.

-
- ❑ if parity is enabled for communication, the UART of the transmitting device adds a parity bit (bit value is 1 for odd number of 1s in the transmitted bit stream and 0 for even number of 1s). .
 - ❑ The UART of the receiving device calculates the parity of the bits received and compares it with the received parity bit for error checking.
 - ❑ The UART of the receiving device discards the 'Start', 'Stop' and 'Parity' bit from the received bit stream and converts the received serial bit data to a word.
 - ❑ For proper communication, the 'Transmit line' of the sending device should be connected to the 'Receive line' of the receiving device.

- ❑ The figure illustrates the same
- ❑ In addition to the serial data transmission function, UART provides hardware handshaking signal support for controlling the serial data flow.
- ❑ UART chips are available from different semiconductor manufacturers.
- ❑ National Semiconductor's 8250 UART chip is considered as the standard setting UART.
- ❑ It was used in the original IBM PC.



1 – Wire Interface

- ❑ **1-Wire Interface:** is an asynchronous half-duplex communication protocol developed by Maxim Dallas Semiconductor.
- ❑ It is also known as *Dallas 1-Wire® protocol*.
- ❑ It makes use of only a single signal line (wire) called *DQ (Data Query)* for communication and follows the master-slave communication model.
- ❑ One of the key features of 1-wire bus is that it allows power to be sent along the signal wire as well.
- ❑ The 1-wire slave devices incorporate an internal capacitor (typically of the order of 800 pF) to power the device from the signal line.
- ❑ The 1-wire interface supports a single master and one or more slave devices on
- ❑ the bus.

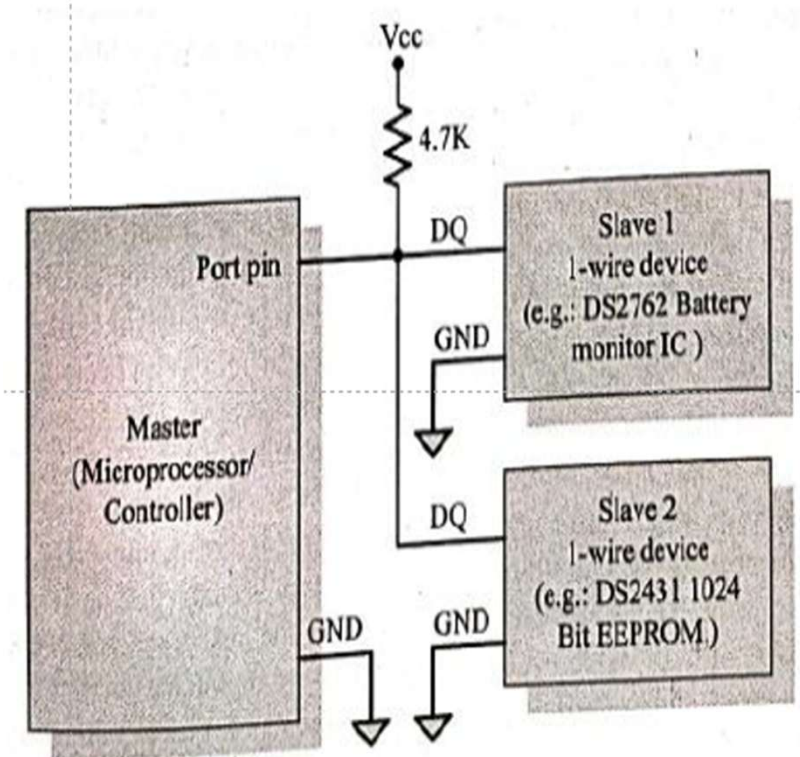
❑ The bus interface diagram shown in the following Figure illustrates the connection of master and slave devices on the 1-wire bus.

❑ Every 1-wire device contains a globally unique 64-bit identification number stored within it.

❑ This unique identification number can be used for addressing individual devices present on the bus in case there are multiple slave devices

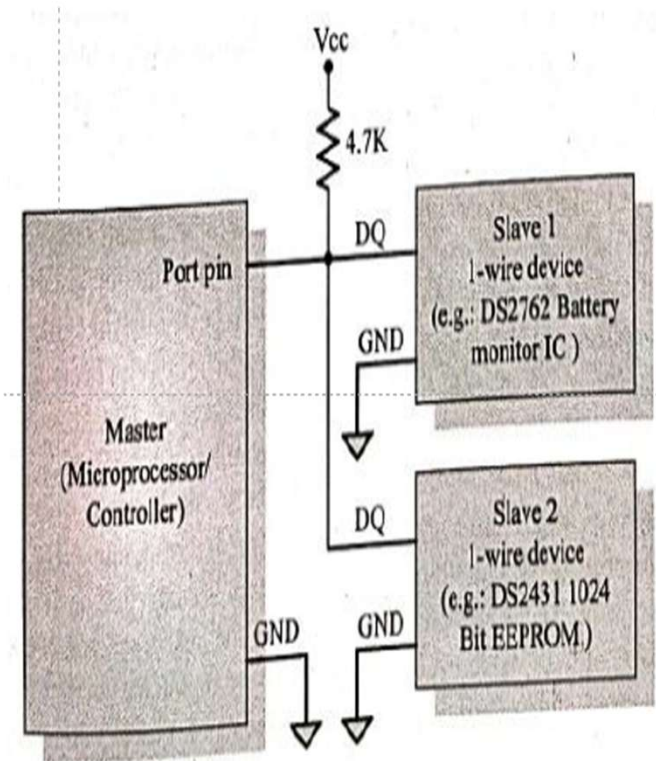
❑ The identifier has three parts: an 8-bit family code, a 48-bit serial number and an 8-bit CRC computed from the first 56-bits.

❑ (CRC: Cyclic Redundancy Check)



❑ **The sequence of operation** for communicating with a 1-wire slave device is listed below:

1. The master device sends a 'Reset' pulse on the 1-wire bus.
2. The slave device(s) present on the bus respond with a 'Presence' pulse.
3. The master device sends a ROM command (Net Address Command followed by the 64-bit address of the device). This addresses the slave device(s) to which it wants to initiate a communication.
4. The master device sends a read/ write function command to read/ write the internal memory or register of the slave device.
5. The master initiates a Read data/ Write data from the device or to the device.

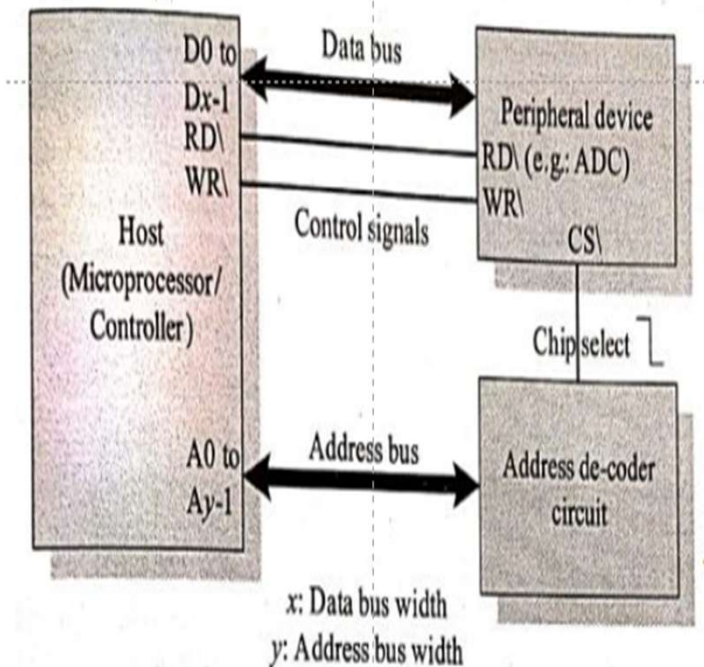


-
- ❑ All communication over the 1-wire bus is master initiated.
 - ❑ The communication over the 1-wire bus is divided into timeslots of 60 microseconds for regular speed mode of operation (16.3Kbps).

Parallel Interface

- ❑ Parallel Interface: The on-board parallel interface is normally used for communicating with peripheral devices which are memory mapped to the host of the system.
- ❑ The host processor/ controller of the embedded system contains a parallel bus and the device which supports parallel bus can directly connect to this bus system.
- ❑ The communication through the parallel bus is controlled by the control signal interface between the device and the host.
- ❑ The 'Control Signals' for communication includes 'Read/ Write' signal and device select signal.
- ❑ The device normally contains a device select line and the device becomes active only when this line is asserted by the host processor.

- ❑ The width of the parallel interface is determined by the data bus width of the host processor. It can be 4-bit, 8-bit, 16-bit, 32-bit or 64-bit, etc.
- ❑ The bus width supported by the device should be same as that of the host processor.
- ❑ Parallel data communication offers the highest speed for data transfer.
- ❑ The bus interface diagram shown in the following Figure, illustrates the interfacing of devices through parallel interface.



External Communication Interfaces

□ The External Communication refers to the different communication channels/buses used by the Embedded system to communicate with the external world.

- ❖ RS-232C and RS-485
- ❖ Universal Serial Bus (USB)
- ❖ IEEE1394 (Fire wire)
- ❖ Infrared (IrDA)
- ❖ Bluetooth(BT)
- ❖ Wi-Fi
- ❖ ZigBee
- ❖ General Packet Radio Service(GPRS), 3G, 4G, LTE

RS-232C AND RS-485:

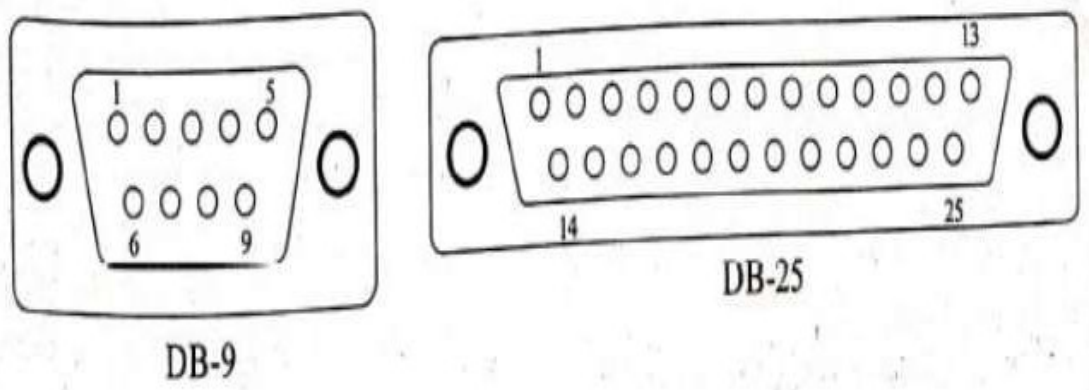
- ❑ RS-232C(Recommended Standard number 232,revision C) from the Electronic Industry Association is a legacy, full duplex, wired, asynchronous serial communication interface.
- ❑ The RS-232 interface is developed by the Electronics Industries Association(EIA) during the early 1960s.
- ❑ RS-232 extends the UART communication signals for external data communication.
- ❑ UART uses the standard TTL/CMOS logic for bit transmission; whereas RS-232 follows the EIA standard for bit transmission.

-
- ❑ As per the EIA standard, a logic '0' is represented with voltage between +3 and +25V and a logic '1' is represented with voltage between -3 and -25V.
 - ❑ In EIA standard, logic '0' is known as 'Space' and logic '1' as 'Mark'.
 - ❑ The RS-232 interface defines various handshaking and control signals for communication apart from the “Transmit” and “Receive” signal lines for data communication.

- ❑ RS-232 supports two different types of connectors:
- ❑ DB-9 : 9-Pin connector.
- ❑ DB-25: 25-Pinconnector.
- ❑The following Figure illustrates the connector details for DB-9 and DB-25.



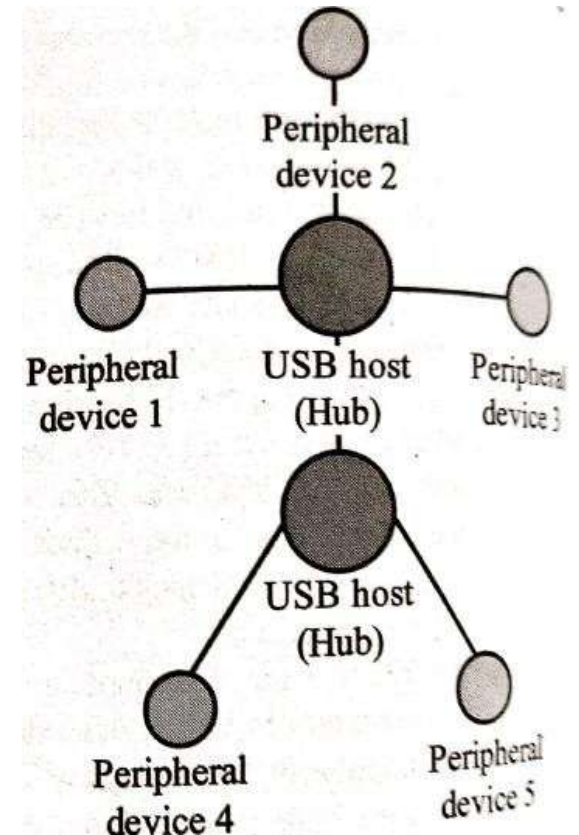
shutterstock.com · 1602241555



UNIVERSAL SERIAL BUS (USB):

- ❑ **Universal Serial Bus (USB):** Universal Serial Bus is a wired high speed serial bus for data communication.
- ❑ The first version of USB (USB 1.0) was released in 1995 and was created by the USB core group members consisting of Intel, Microsoft, IBM, Compaq, Digital and Northern Telecom.
- ❑ The USB communication system follows a star topology with a USB host at the centre and one or more USB peripheral devices/ USB hosts connected to it.
- ❑ A USB 2.0 host can support connections up to 127, including slave peripheral devices and other USB hosts

- ❑ The following Figure illustrates the **star topology for USB device connection**.
- ❑ **USB transmits data in packet format**
- ❑ **Each data packet has a standard format.**
- ❑ **The USB communication is a host initiated one.**
- ❑ **The USB host contains a host controller** which is responsible for controlling the data communication, including establishing connectivity with USB slave devices, packetizing and formatting the data.



-
- ❑ The physical connection between a USB peripheral device and master device is established with a USB cable.
 - ❑ The USB cable in USB 2.0 supports communication distance of up to 5 meters.
 - ❑ The USB 2.0 standard uses two different types of connector at the ends of the USB cable for connecting the USB peripheral device and host device.

1. Type 'A' Connector



2. Type 'B' Connector



-
- ❑ 'Type A' connector is used for upstream connection (connection with host).
 - ❑ Type B connector is used for downstream connection (connection with slave device).
 - ❑ The USB connector present in desktop PCs or laptops are examples for 'Type A' USB connector.
 - ❑ Both Type A and Type B connectors contain 4 pins for communication.

USB DATA TRANSFERS

□ USB supports 4 different types of data transfers, namely:

1. Control
2. Bulk
3. Isochronous
4. Interrupt.

□ **Control transfer** is used by USB system software to **query, configure and issue commands** to the USB device.

□ **Bulk transfer** is used for **sending a block of data** to a device.

❖ Bulk transfer supports **error checking and correction**.

❖ Transferring **data to a printer** is an example for bulk transfer.

❑ *Isochronous data transfer* is used for real-time data communication.

❖ In **Isochronous transfer**, data is transmitted as streams in real-time.

❖ Isochronous transfer doesn't support error checking and retransmission of data in case of any transmission loss.

❖ All streaming devices like audio devices and medical equipment for data collection make use of the isochronous transfer.

❑ *Interrupt transfer* is used for **transferring small amount of data**.

❖ Interrupt transfer mechanism makes use of “**polling technique**” to see whether the USB device has any data to send.

❖ Devices like Mouse and Keyboard, which transmits fewer amounts of data, uses Interrupt transfer.

IEEE 1394 (FIREWIRE):

- ❑ *IEEE 1394 (Firewire): IEEE 1394* is a wired isochronous high speed serial communication bus. It is also known as *High Performance Serial Bus (HPSB)*
- ❑ FireWire, also called IEEE 1394 or I. LINK, high-speed computer data-transfer interface that was used to **connect personal computers, audio and video devices, and other professional and consumer electronics.**
- ❑ IEEE 1394 supports peer-to-peer connection and point-to-multipoint communication allowing 63 devices to be connected on the bus in a tree topology.
- ❑ IEEE 1394 is a **wired serial interface** and it can support a cable length of up to **15 feet** for interconnection.
- ❑ 1394 is a **popular communication interface** for connecting embedded devices like **Digital Camera, Camcorder, Scanners** to **desktop computers** for data transfer and storage

Infrared Light Emitting Diode

- ❑ Infrared **Light Emitting Diode (LED)** is the IR source for transmitter and at the receiving end a **photodiode** acts as the receiver.
- ❑ Both transmitter and receiver unit will be present in each device supporting **IrDA** communication for bidirectional data transfer.
 - ❖ Such IR units are known as '**Transceiver**'.
- ❑» Certain devices like a TV remote control always require unidirectional communication and so they contain either the transmitter or receiver unit (The remote control unit contains the transmitter unit and TV contains the receiver unit).

BLUETOOTH (BT):

- ❑ **Bluetooth (BT):** Bluetooth is a low cost, low power, short range wireless technology for data and voice communication.
- ❑ Bluetooth was first proposed by 'Ericsson' in 1994.
- ❑ Bluetooth operates at *2.4GHz of the Radio Frequency spectrum* and uses the *Frequency Hopping Spread Spectrum (FHSS)* technique for communication.
- ❑ Literally it supports a data rate of up to *1Mbps* and a range of approximately **30 to 100 feet (version dependent)** for data communication.

-
- ❑ Each Bluetooth device will have a 48-bit unique identification number.
 - ❑ Bluetooth communication follows packet based data transfer.
 - ❑ Bluetooth supports *point-to-point (device to device)* and *point-to-multipoint (device to multiple device broadcasting)* wireless communication.
 - ❑ The *point-to-point* communication follows the *master-slave* relationship.
 - ❑ A Bluetooth device can function as either master or slave.
 - ❑ When a network is formed with one Bluetooth device as master and more than one device as slaves, it is called a *Piconet*.
 - ❑ A *Piconet* supports a maximum of **seven slave devices**.

WI-FI

- ❑ **Wi-Fi:** *Wi-Fi or Wireless Fidelity* is the popular wireless communication technique for networked communication of devices.
- ❑ Wi-Fi follows the **IEEE 802.11** standard.
- ❑ Wi-Fi is intended for *network communication* and supports *Internet Protocol (IP)* based communication.
- ❑ It is essential to have device identities in a multi-point communication to address specific devices for data communication.
- ❑ In an *IP based communication* each device is identified by an *IP address*, which is unique to each device on the network.
- ❑ Wi-Fi based communications require an *intermediate agent* called *Wi-Fi router/Wireless Access point* to manage the communications.

❑ Wi-Fi enabled devices contain a *wireless adaptor* for transmitting and receiving data in the form of radio signals through an antenna.

❑ The hardware part of it is known as *Wi-Fi Radio*.

❑ The following Figure illustrates the *typical interfacing of devices in a Wi-Fi network*.



ZIGBEE:

- ❑ **ZigBee:** ZigBee is a *low power, low cost, wireless network communication protocol* based on the **IEEE 802.15.4-2006** standard.
- ❑ ZigBee is a Personal Area Network task group with low rate task group 4.
- ❑ It is a technology of home networking.
- ❑ ZigBee is a technological standard created for controlling and sensing the network.
- ❑ The ZigBee specifications support a robust mesh network containing multiple nodes.
- ❑ This networking strategy makes the network reliable by permitting messages to travel through a number of different paths to get from one node to another.

TYPES OF ZIGBEE DEVICES:

Types of ZigBee Devices:

❑ Zigbee Coordinator Device:

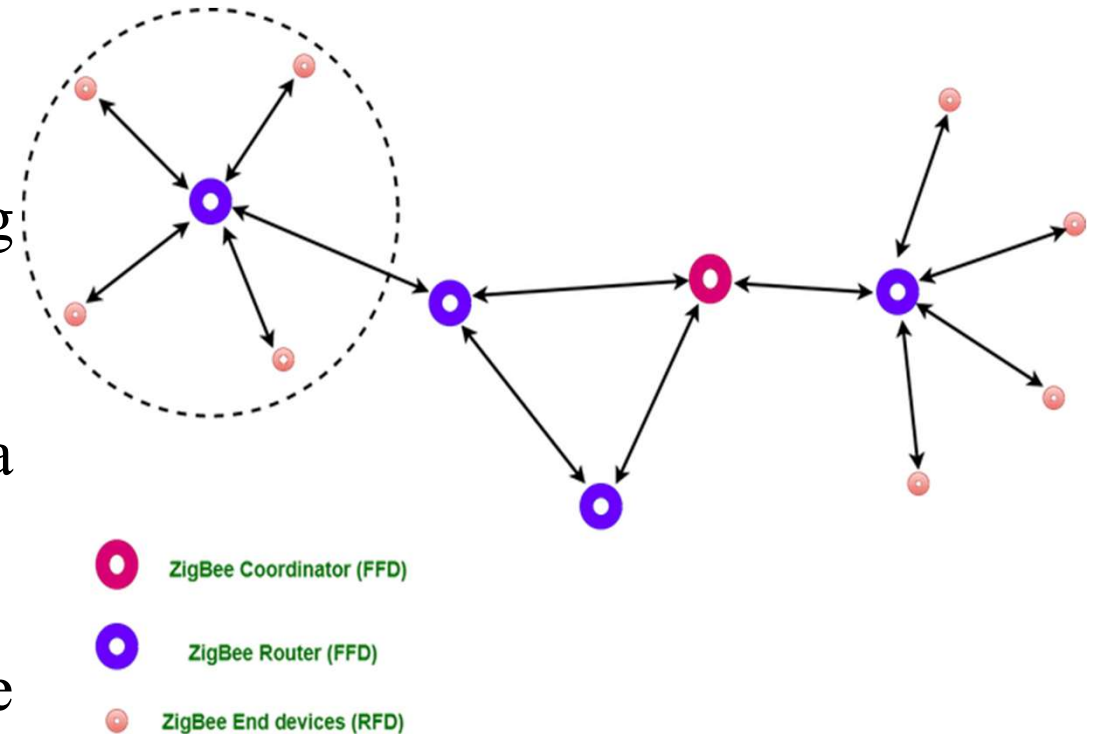
- ❖ It communicates with routers.
- ❖ This device is used for connecting the devices.

❑ Zigbee Router:

- ❖ It is used for passing the data between devices.

❑ Zigbee End Device:

- ❖ It is the device that is going to be controlled.



(GPRS), 3G, 4G, LTE

❑ General Packet Radio Service (GPRS), 3G, 4G, LTE:

❑ General Packet Radio Service is a communication technique for transferring data over a mobile communication network like GSM.

❑ Data is sent as packets in GPRS communication.

❑ The transmitting device splits the data into several related packets.

❑ At the receiving end the data is re-constructed by combining the received data packets.

❑ GPRS supports *Internet Protocol (IP)*, *Point to Point Protocol (PPP)* and *X.25 protocols* for communication.

SERVICES OFFERED BY GPRS

□ Services Offered:

- ❖ *SMS messaging and broadcasting*
- ❖ *Push-to-talk over cellular*
- ❖ *Instant messaging and presence*
- ❖ *Multimedia messaging service*
- ❖ *Point-to-Point and Point-to-Multipoint services*

BENEFITS OF GPRS:

❑ *Mobility:*

- ❖ The capacity to keep up consistent voice and information interchanges while moving.

❑ *Cost Efficient:*

- ❖ Communication via GPRS is cheaper than through the regular GSM network.

❑ *Immediacy:*

- ❖ Allows customers to obtain connectivity when needed, regardless of location and without a lengthy login session.

❑ *Localization:*

- ❖ Enables customers to acquire data applicable to their present area.

❑ *Easy Billing:*

- ❖ GPRS packet transmission offers an easier to use billing than that offered by circuit switched administrations.

EMBEDDED FIRMWARE

- ❑ **Embedded firmware** refers to the **control algorithm** (Program instructions) and or the **configuration settings** that an embedded system developer dumps into the code (Program) memory of the embedded system.
 - ❑ It is an un-avoidable part of an embedded system.
 - ❑ There are various methods available for developing the embedded firmware.
 - ❑ **They are listed below:**
1. **Write the program in high level languages like Embedded C/ C++ using an Integrated Development Environment (IDE)**
 - ❖ The IDE will contain a editor, compiler, linker, debugger, simulator, etc.
 - ❖ IDEs are different for different family of processors/ controllers.
 - ❖ For example, **Keil** microvision3 IDE is used for all family member of 8051 microcontroller, since it contains the generic 8051 compiler C51.

❑ 2. Write the program in Assembly language using the instructions supported by your application's target processor/ controller.

❑ The instruction set for each family of processor/ controller is different and the program written in either of the methods given above should be converted into a processor understandable machine code before loading it into the program memory.

❑ The process of converting the program written in either a high level language or processor/ controller specific Assembly code to machine readable binary code is called '*HEX File Creation*'.

-
- ❑ The methods used for 'HEX File Creation' is different depending on the programming techniques used.
 - ❑ If the program is written in Embedded C/ C++ using an IDE, the cross compiler included in the IDE converts it into corresponding processor/ controller understandable 'HEX File'.
 - ❑ If you are following the Assembly language based programming technique, you can use the utilities supplied by the processor/ controller vendors to convert the source code into 'HEX File'.
 - ❑ Also third party tools are available, which may be of free of cost, for this conversion.

OTHER SYSTEM COMPONENTS

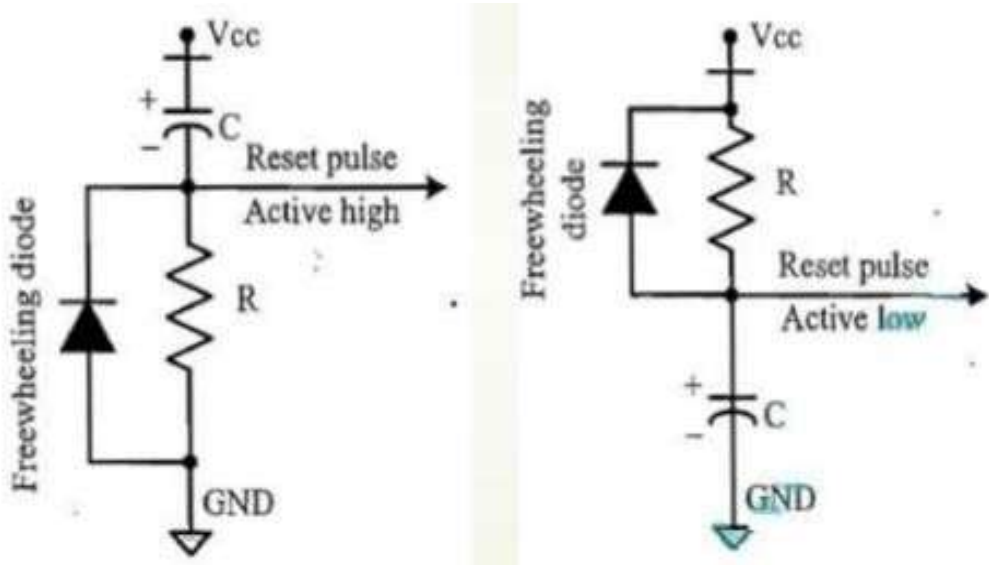
- ❑ The *other system components* refer to the components/ circuits/ ICs which are necessary for the proper functioning of the embedded system.
- ❑ Some of these circuits may be essential for the proper functioning of the processor/ controller and firmware execution.
- ❑» Watchdog timer, Reset IC, brown-out protection IC etc., are examples of circuits/ ICs which are essential for the proper functioning of the processors/ controllers.

RESET CIRCUIT:

- ❑ *Reset Circuit:* The *reset circuit* is essential to ensure that the device is not operating at a voltage level where the device is not guaranteed to operate, during system power ON.
- ❑ The reset signal brings the internal registers and the different hardware systems of the processor/ controller to a known state and starts the firmware execution from the reset vector.
- ❑» The reset signal can be either active high or active low.
- ❑ Since the processor operation is synchronized to a clock signal, the reset pulse should be wide enough to give time for the clock oscillator to stabilize before the internal reset state starts.

» The following Figure illustrates a resistor capacitor based passive reset circuit for active high and low configurations.

The reset pulse width can be adjusted by changing the resistance value R and capacitance value C .



BROWN-OUT PROTECTION CIRCUIT

- ❑ » ***Brown-out Protection Circuit:*** *Brown-out protection circuit* prevents the processor/ controller from unexpected program execution behavior when the supply voltage to the processor/ controller falls below a specified voltage.
- ❑ » It is essential for battery powered devices since there are greater chances for the battery voltage to drop below the required threshold.
- ❑ The processor behavior may not be predictable if the supply voltage falls below the recommended operating voltage. It may lead to situations like data corruption.
- ❑ » A brown-out protection circuit holds the processor/ controller in reset state, when operating voltage falls below the threshold, until it rises above the threshold voltage.

OSCILLATOR UNIT:

- ❑ ***Oscillator Unit:*** A microprocessor/ microcontroller is a digital device made up of digital combinational and sequential circuits.
- ❑ The instruction execution of a microprocessor/ controller occurs in synchronization with a clock signal.
- ❑ The *oscillator unit* of the embedded system is responsible for generating the precise clock for the processor.
- ❑» Certain processors/ controllers integrate a **built-in oscillator unit** and simply require a quartz crystal for producing the necessary clock signals.
- ❑» Certain devices may not contain built-in oscillator unit and require the clock pulses to be generated and supplied externally.

REAL-TIME CLOCK (RTC):

- Real-Time Clock (RTC):*** *Real-Time Clock* is a system component responsible for keeping track of time.
- RTC holds information like current time (In hours, minutes and seconds) in 12-hour/ 24-hour format, date, month, year, day of the week, etc. and supplies timing reference to the system.
- RTC is intended to function even in the absence of power.

RTC CTD..

- ❑ RTCs are available in the form of Integrated Circuits from different semiconductor manufacturers like Maxim/Dallas, ST Microelectronics etc.
- ❑ The RTC chip contains a microchip for holding the time and date related information and backup battery cell for functioning in the absence of power, in a single IC package.
- ❑ The RTC chip is interfaced to the processor or controller of the embedded system.

WATCHDOG TIMER

Watchdog Timer:

In desktop Windows systems, if we feel our application is behaving in an abnormally or if the system hangs up, we have the '*Ctrl + Alt + Del*' to come out of the situation.

What it happens to embedded system?

» We have a watchdog to monitor the firmware execution and reset the system processor/ microcontroller when the program execution hangs up.

A *watchdog timer*, or simply a *watchdog*, is a hardware timer for monitoring the firmware execution.

-
- » Depending on the internal implementation, the watchdog timer increments or decrements a free running counter with each clock pulse and generates a reset signal to reset the processor if the count reaches zero for a *down counting* watchdog, or the highest count value for an *up counting* watchdog