UNIT-2
Embedded System Architectures

By
V Radha Haneesha
Department of ECE
Radha.haneesha@svecw.edu.in

TYPICAL EMBEDDED SYSTEMS BLOCK DIAGRAM

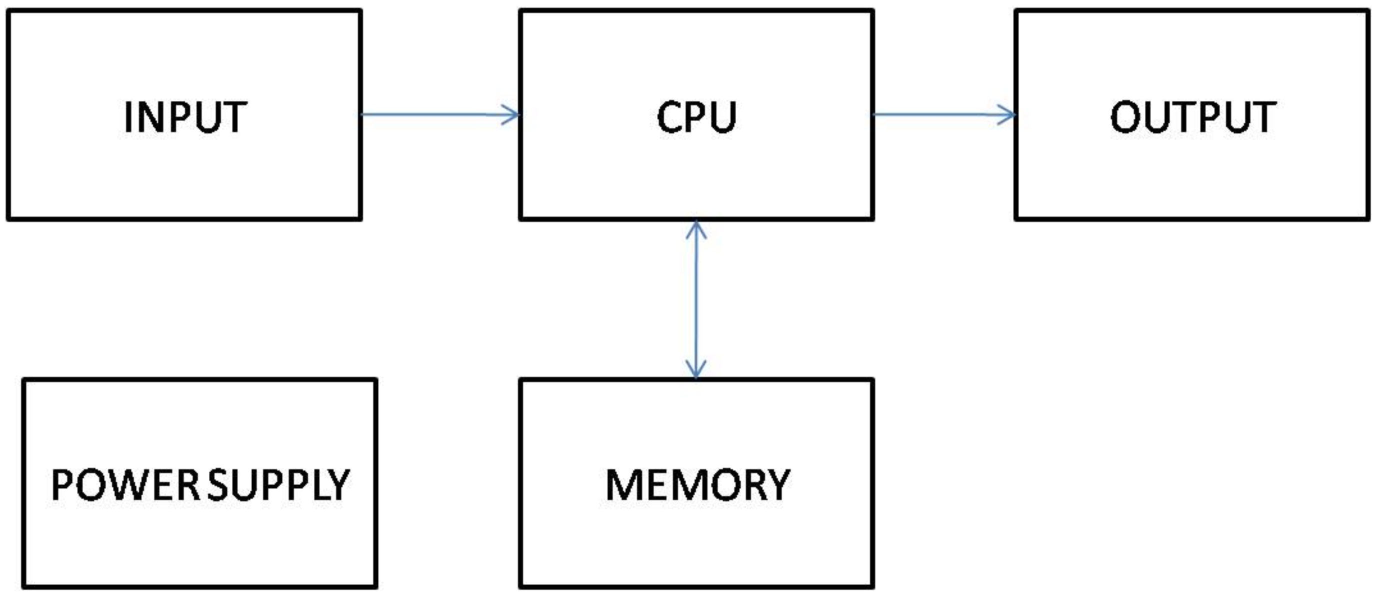


Fig: General Block Diagram Of An Embedded System

- For instance, every embedded system contains a processor and software that runs on the processor.
- The processor may be a microcontroller or a microprocessor.
- Also, in order to have software there must be a place to store the executable code and temporary storage for run-time data manipulations which will take the form of ROM and RAM respectively.
- In case of small memories they may be contained in the same chip as the processor. Generally microcontrollers will have such arrangements. Otherwise one or both types of memory will reside in external memory chips. Furthermore, all embedded systems also contain some type of inputs and outputs. Inputs to the system are generally sensors and probes

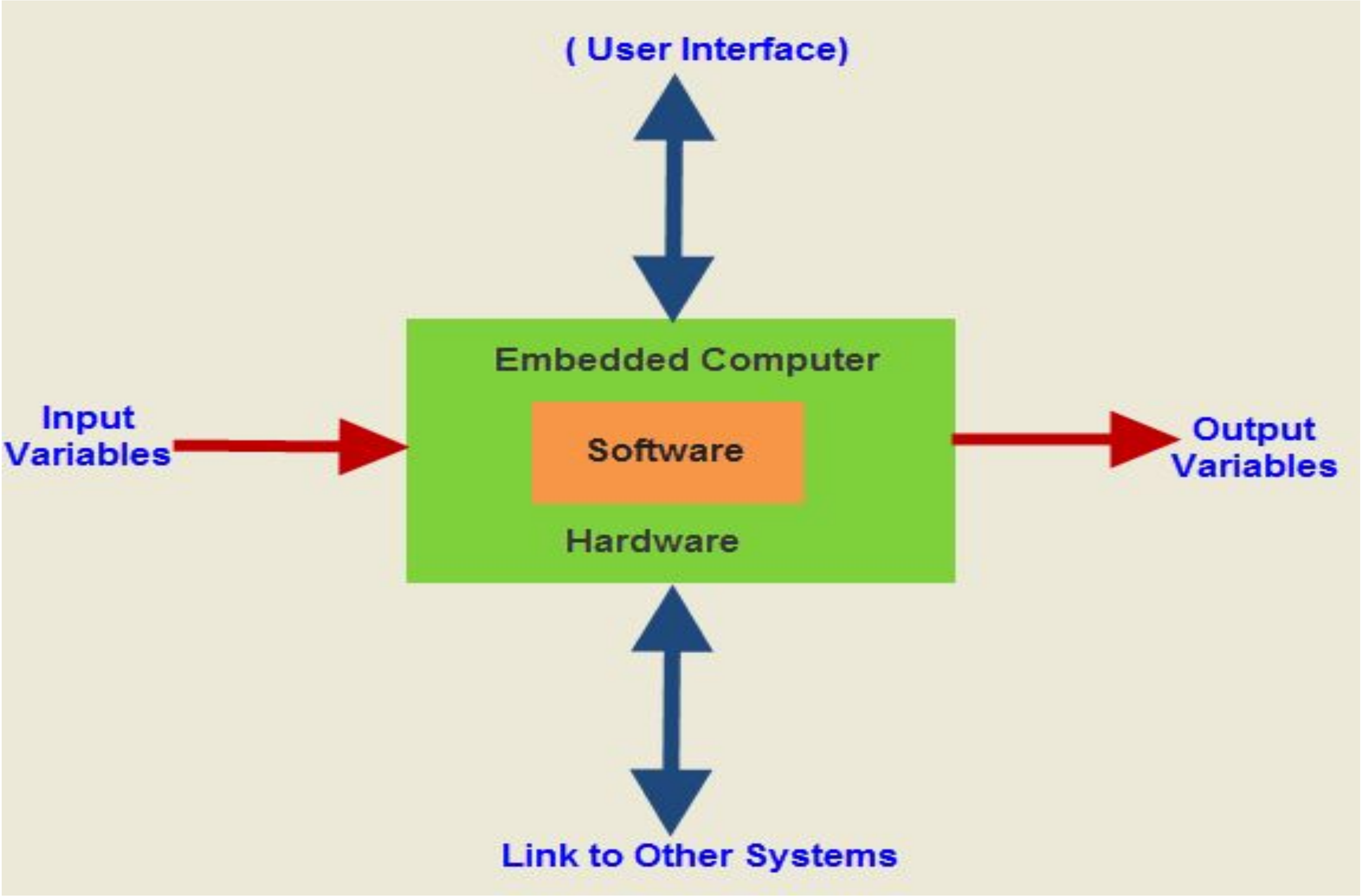


Fig: Block Diagram Of An Embedded System

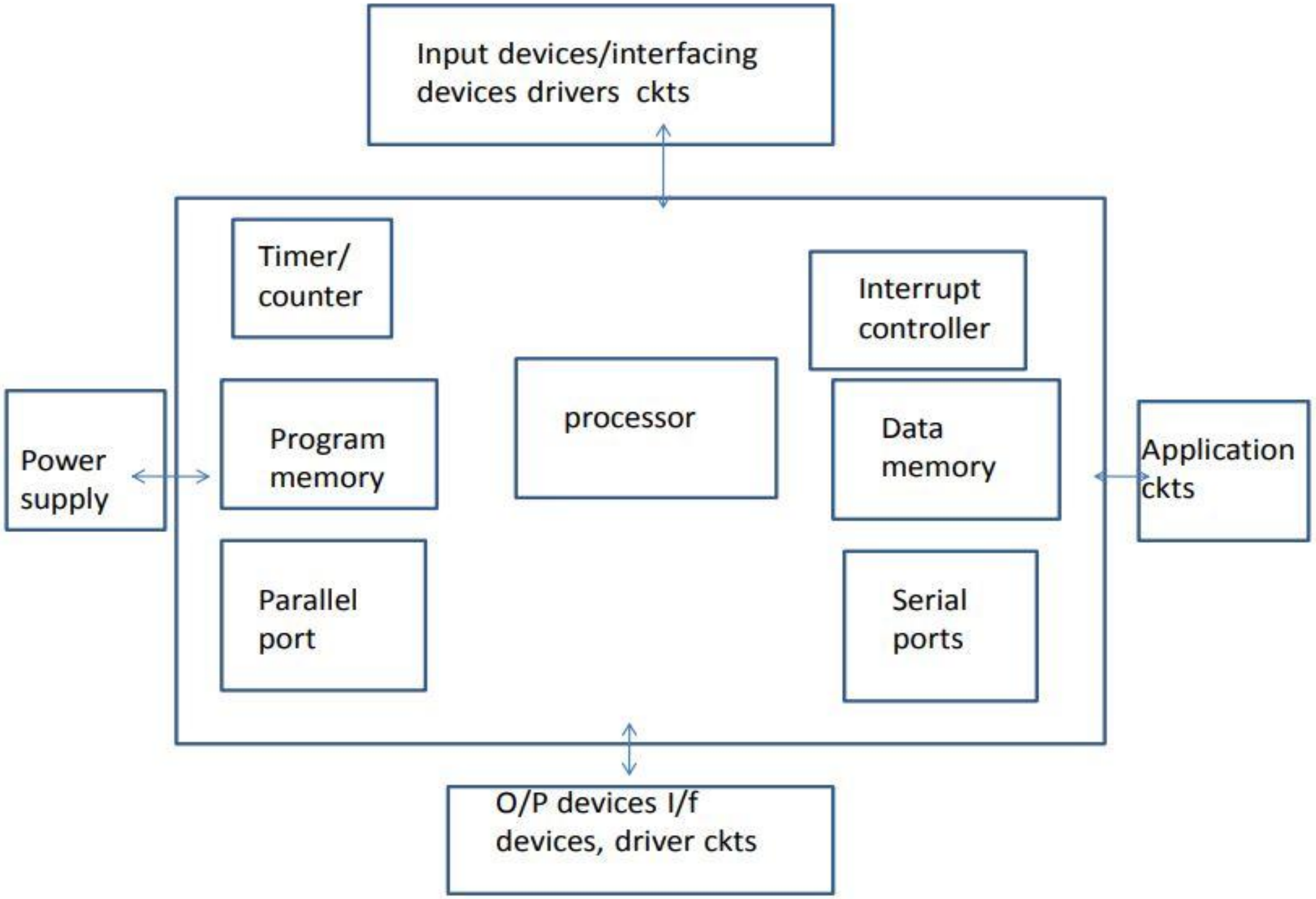


Fig: Block Diagram Of An Embedded System

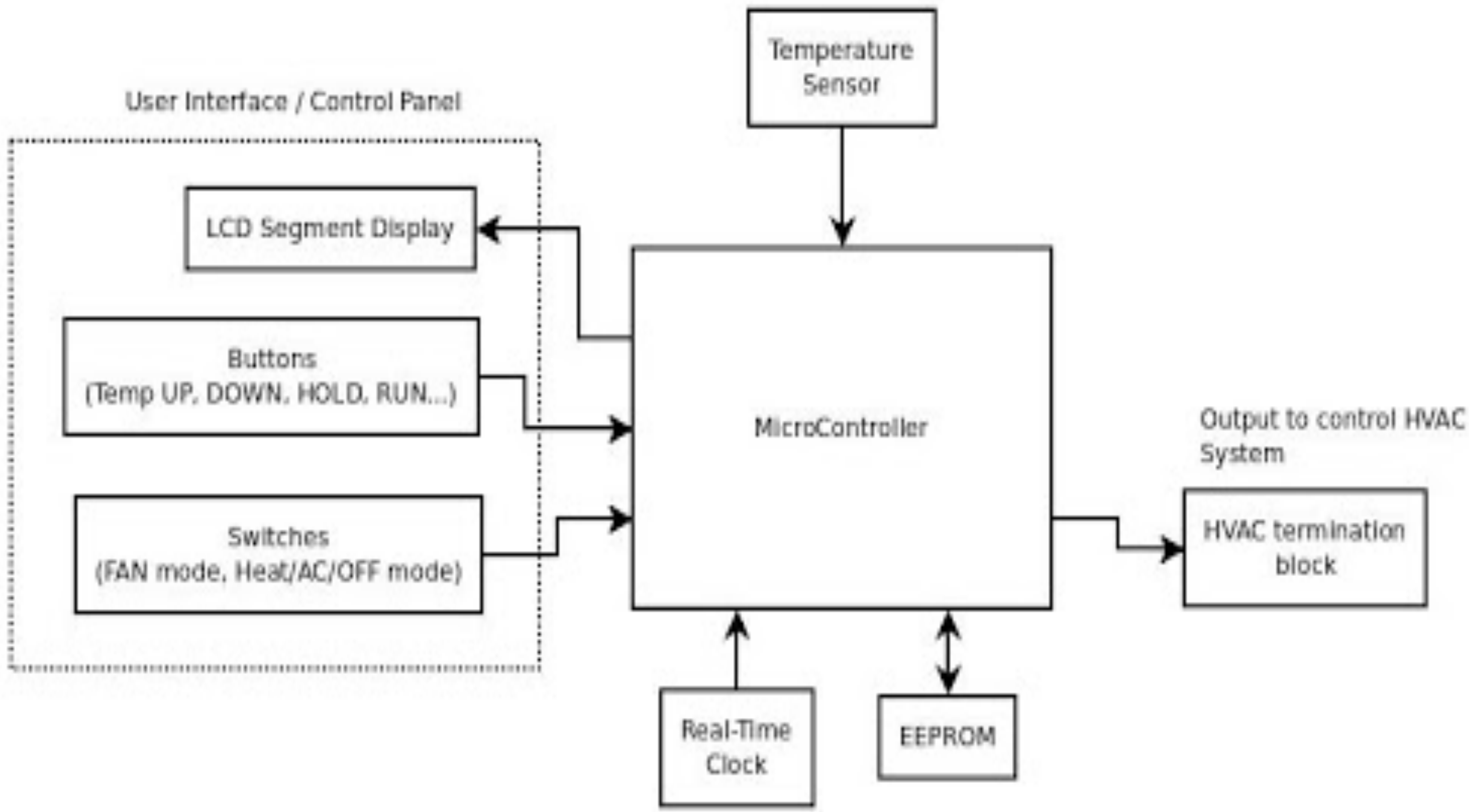


Fig : Block Diagram Of An Embedded System

The typical embedded systems block diagram:

- Memory
- Sensors and actuators
- Communication interface
- Embedded firmware
- other system components

Memory

- On an Embedded System, memory is at a premium. Some chips, particularly embedded VLSI chips, and low-end microprocessors may only have a small amount of RAM "on board" (built directly into the chip), and therefore their memory is not expandable.
- Other embedded systems have a certain amount of memory, and have no means to expand. In addition to RAM, some embedded systems have some nonvolatile memory, in the form of miniature magnetic disks, FLASH memory expansions, or even various 3rd-party memory card expansions.

- **memory upgrade on an embedded system may cost more than the entire system itself.**
- An embedded systems programmer, therefore, needs to be very much aware of the memory available, and the memory needed to complete a task.

- There are typically 4 distinct addressable areas, each one implemented with a different technology:

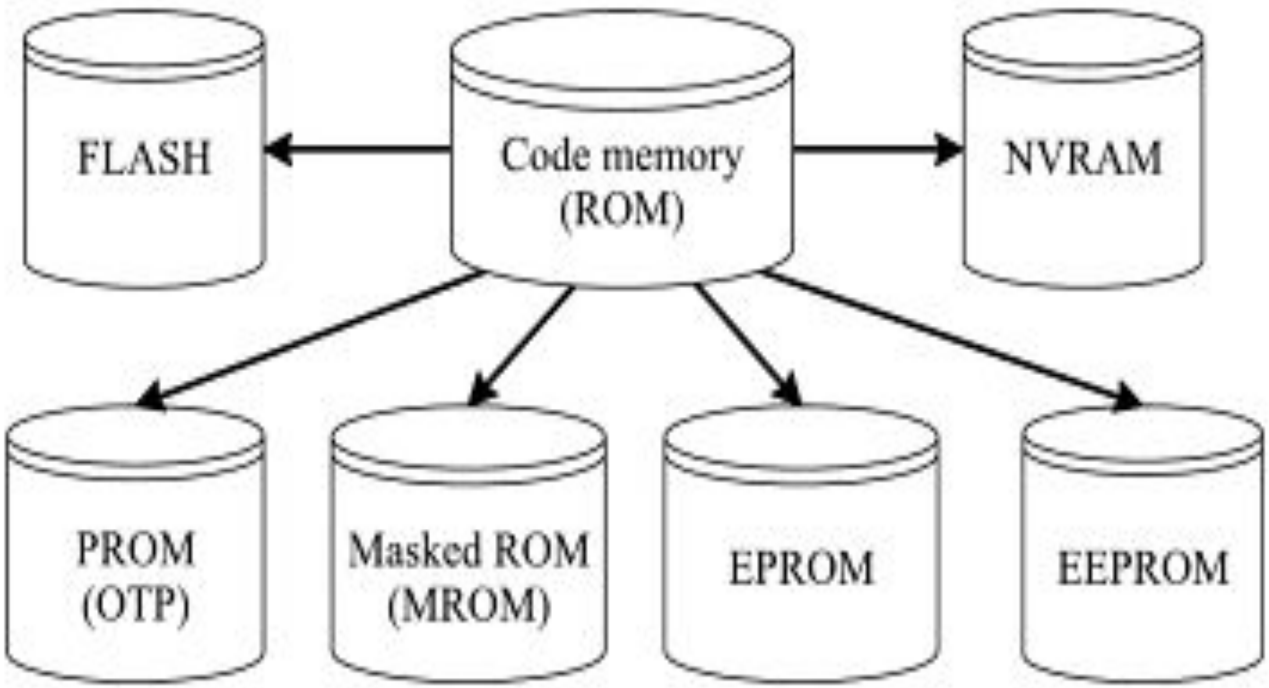
  - RAM
  - ROM
  - EEPROM
  - I/O

- Program memory (which holds the programs you write), often called ROM .While your program is running, it is impossible to change any of the data in program memory. But at least when the power comes back on, it's all still there.
- RAM, which holds the variables and stack. (Initial values for variables are copied from ROM). Forgets everything when power is lost.

- EEPROM. Used kind of like the hard drive in a personal computer, to store settings that might change occasionally, and that need to be remembered next time it starts up.
- I/O. This is really the entire point of a microcontroller, where it is called as input memory and output memory .

- Most CPUs used in desktop machines have a "memory management unit" (MMU). The MMU handles virtual memory, protects regions of memory used by the OS from untreated programs.
- Most embedded systems do not have a MMU.

ROM

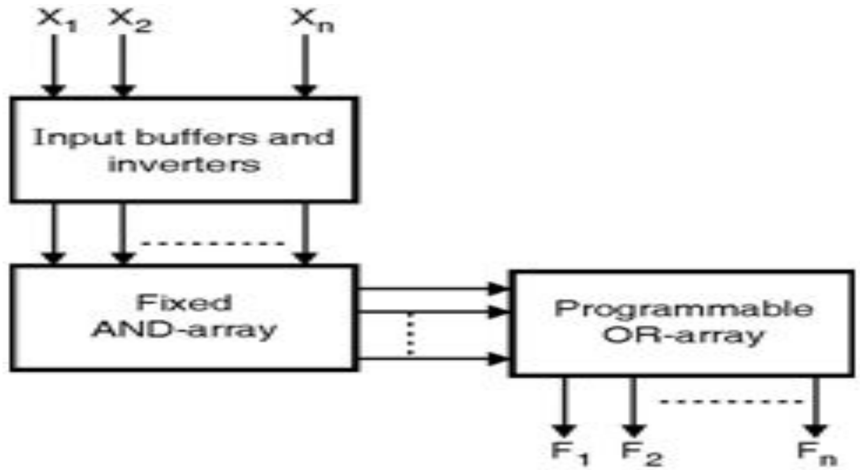

FLASH
Code memory
(ROM)
NVRAM
PROM
(OTP)
Masked ROM
(MROM)
EPROM
EEPROM

- **Mask ROM** (**MROM**) is a type of read-only memory (ROM) whose contents are programmed by the integrated circuit manufacturer (rather than by the user). The terminology *mask* comes from integrated circuit fabrication, where regions of the chip are masked off during the process of photolithography.

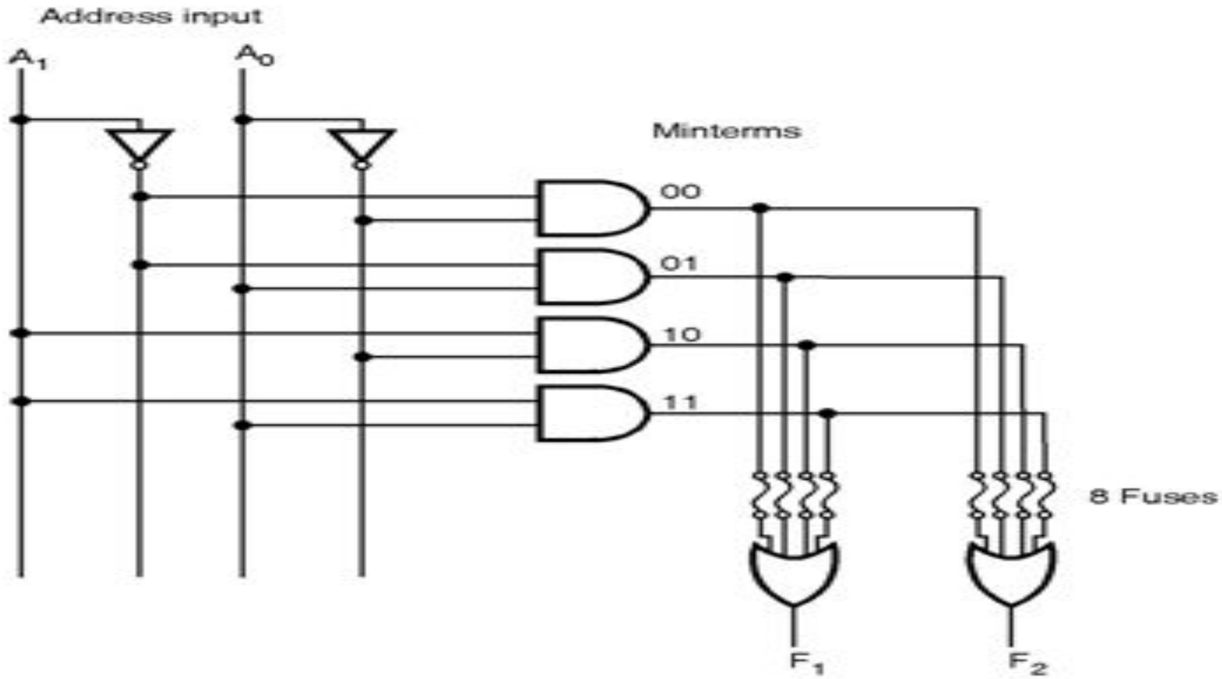
- The main advantage of mask ROM is its cost. Per bit, mask ROM is more compact than any other kind of semiconductor memory. Since the cost of an integrated circuit strongly depends on its size, mask ROM is significantly cheaper than any other kind of semiconductor memory.
- However, the one-time masking cost is high and there is a long turn-around time from design to product phase. Design errors are costly: if an error in the data or code is found, the mask ROM is useless and must be replaced in order to change the code or data.
- As of 2003, four companies produce most such mask ROM chips: Samsung Electronics, NEC Corporation, Oki Electric Industry, and Macronix.

PROM

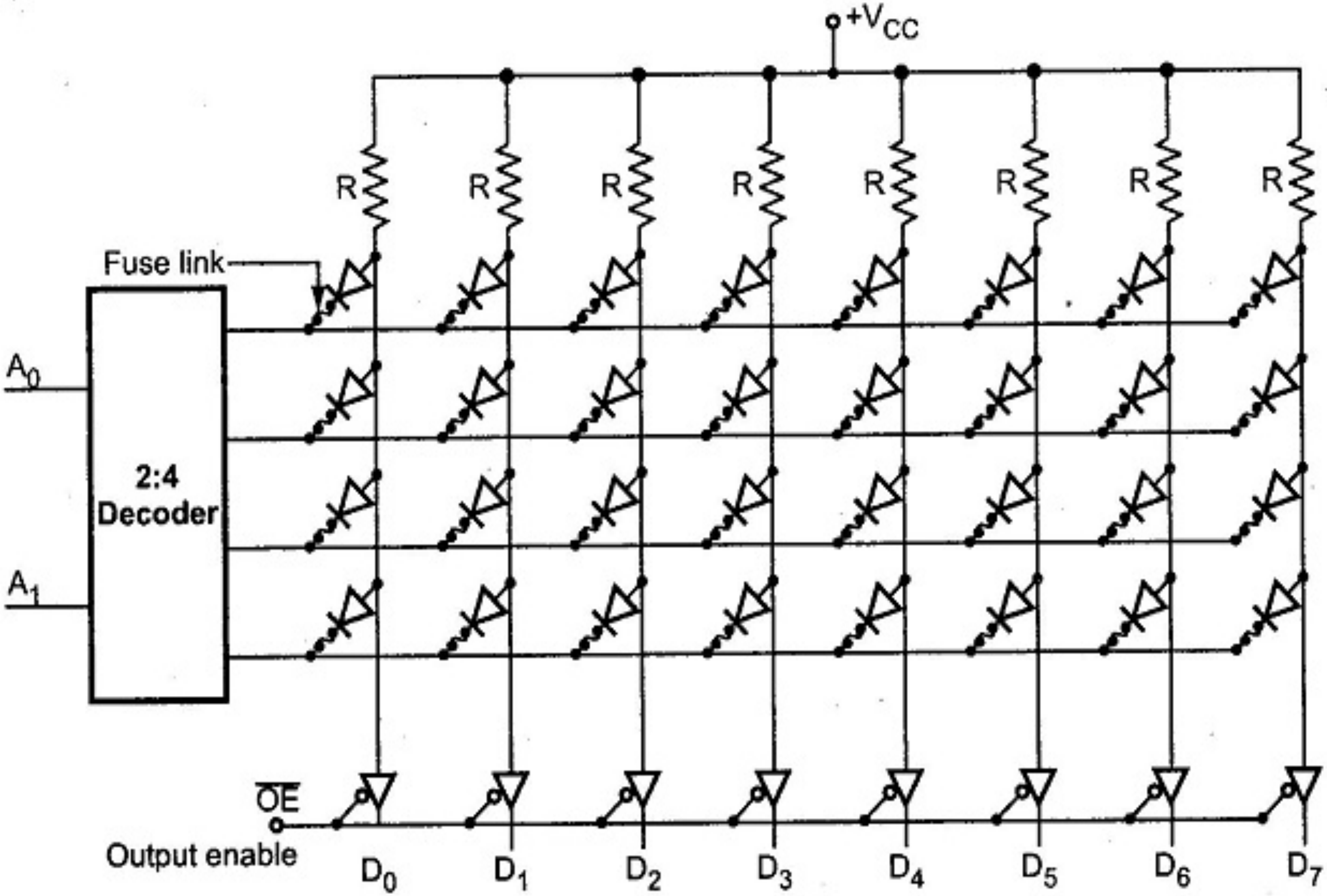
- A **programmable read-only memory** (**PROM**) is a form of digital memory where the setting of each bit is locked by a fuse or antifuse. (eFUSEs can also be used) It is one type of ROM (read-only memory). The data in them are permanent and cannot be changed.
- PROMs are used in digital electronic devices to store permanent data, usually low level programs such as firmware or microcode. The key difference from a standard ROM is that the data is written into a ROM during manufacture, while with a PROM the data is programmed into them after manufacture.



X_1
X_2
X_n
Input buffers and inverters
Fixed AND-array
Programmable OR-array
F_1
F_2
F_n



Address input
A_1
A_0
Minterms
00
01
10
11
8 Fuses
F_1
F_2



+Vcc
R
Fuse link
A0
A1
2:4
Decoder
OE
Output enable
D0
D1
D2
D3
D4
D5
D6
D7

EPROM

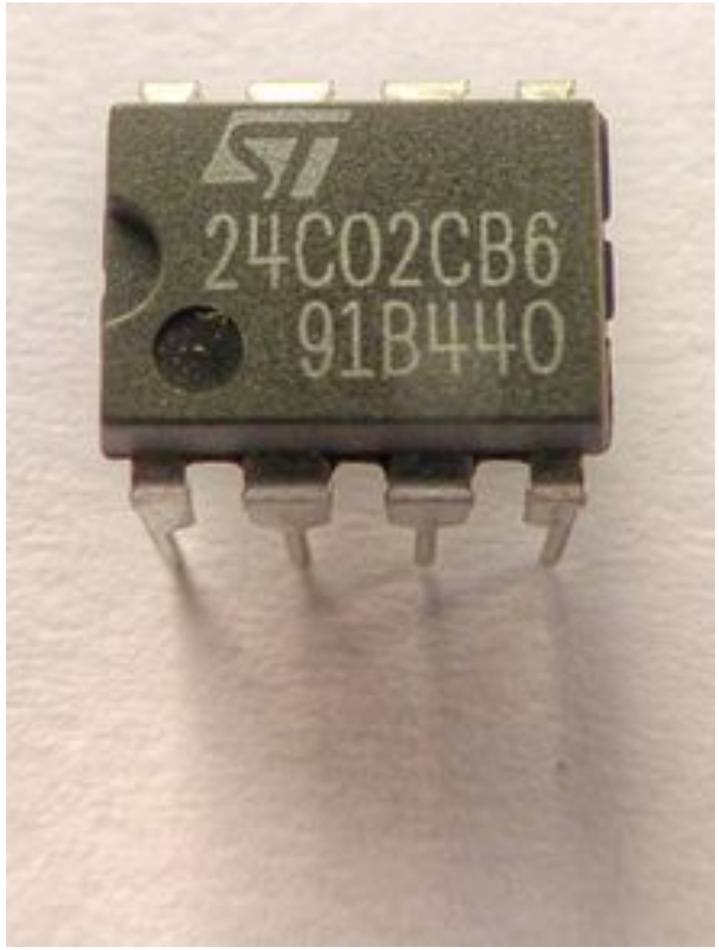
- EPROM (erasable programmable read-only memory) is programmable read-only memory (programmable ROM) that can be erased and re-used. Erasure is caused by shining an intense ultraviolet light through a window that is designed into the memory chip.

EPROM

- Computer memory that can retrieve stored data after a power supply has been turned off and back on is called non-volatile.
- It is an array of floating-gate transistors individually programmed by an electronic device that supplies higher voltages than those normally used in digital circuits.
- EPROMs are easily recognizable by the transparent fused quartz window on the top of the package, through which the silicon chip is visible, and which permits exposure to ultraviolet light during erasing.

EEPROM

- EEPROM (also E^2PROM) stands for electrically erasable programmable read-only memory and is a type of non-volatile memory used in computers, integrated in microcontrollers for smart card and remote keyless systems, and other electronic devices to store relatively small amounts of data but allowing individual bytes to be erased and reprogrammed.



ST Micro serial type EEPROM

- EEPROMs are organized as arrays of floating-gate transistors. EEPROMs can be programmed and erased in-circuit, by applying special programming signals.
- Originally, EEPROMs were limited to single byte operations, which made them slower, but modern EEPROMs allow multi-byte page operations.
- An EEPROM has a limited life for erasing and reprogramming, now reaching a million operations in modern EEPROMs.
- In an EEPROM that is frequently reprogrammed, the life of the EEPROM is an important design consideration.

FLASH MEMORY

- **Flash memory** is an electronic (solid-state) non-volatile computer memory storage medium that can be electrically erased and reprogrammed. The two main types of flash memory are named after the NAND and NOR logic gates.

- Flash memory is a type of floating-gate memory based on EEPROM (electrically erasable programmable read-only memory) technology. Toshiba commercially introduced flash memory to the market in 1987.
- While EPROMs had to be completely erased before being rewritten, NAND-type flash memory may be erased, written and read in blocks (or pages) which are generally much smaller than the entire device. NOR-type flash allows a single machine word (byte) to be written – to an erased location – or read independently

NVRAM

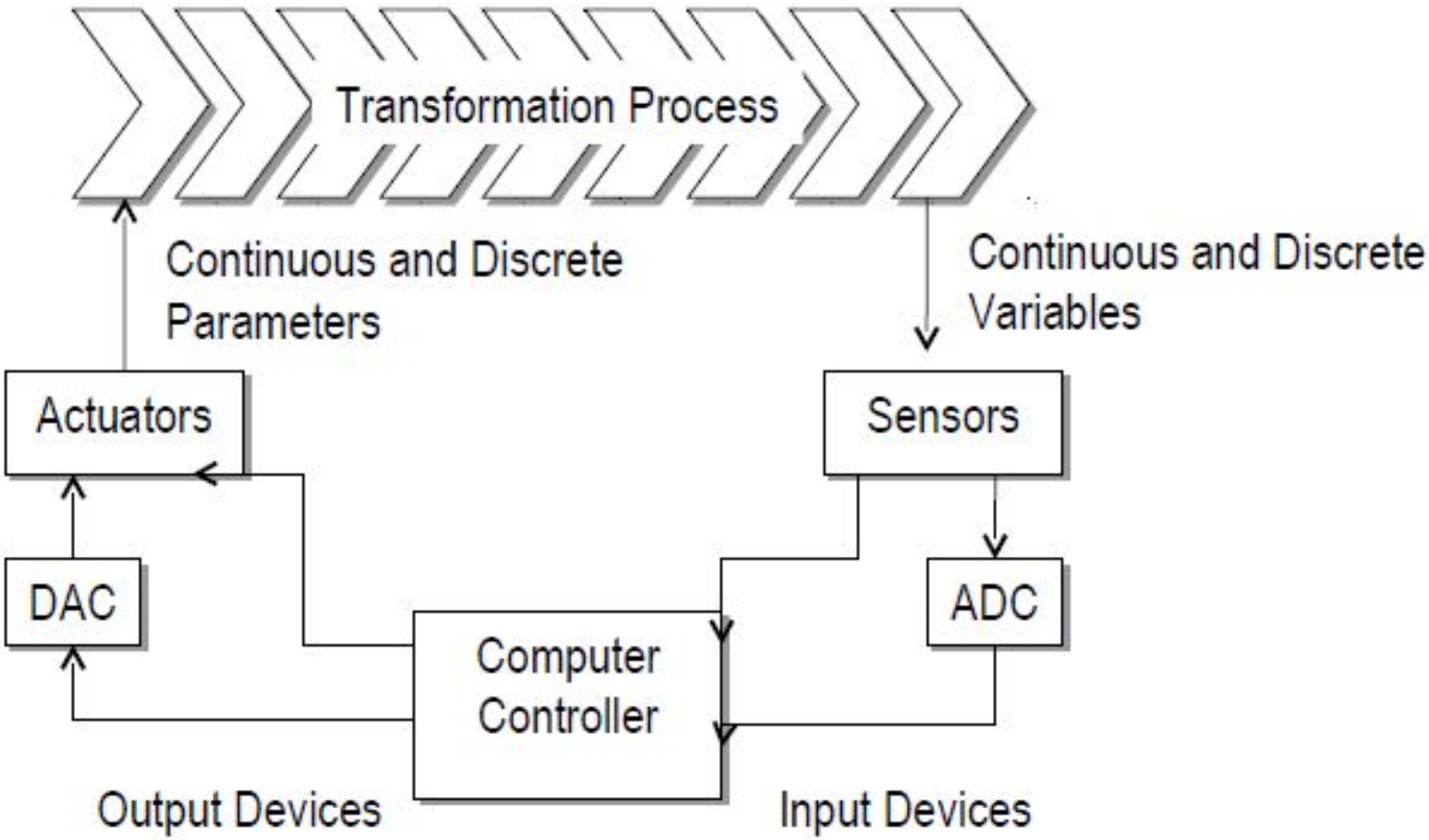
- Stands for "Non-Volatile Random Access Memory." NVRAM is a type of RAM that retains data after the host device's power is turned off. Two common types of NVRAM include SRAM and EEPROM.
- SRAM (pronounced "s-ram") retains data by using an alternative source of power such as a battery. SRAM is often used to store computer hardware settings that need to be maintained when the computer is shut down.

- Common examples include the BIOS settings on Windows computers or the PRAM settings on Macintosh systems. Since SRAM typically uses a battery to retain memory, if the battery dies or is disconnected, the data stored in the SRAM will be lost. Therefore, if BIOS or PRAM settings are not retained after a computer is restarted, it is likely the computer's battery has lost its charge and needs to be replaced.

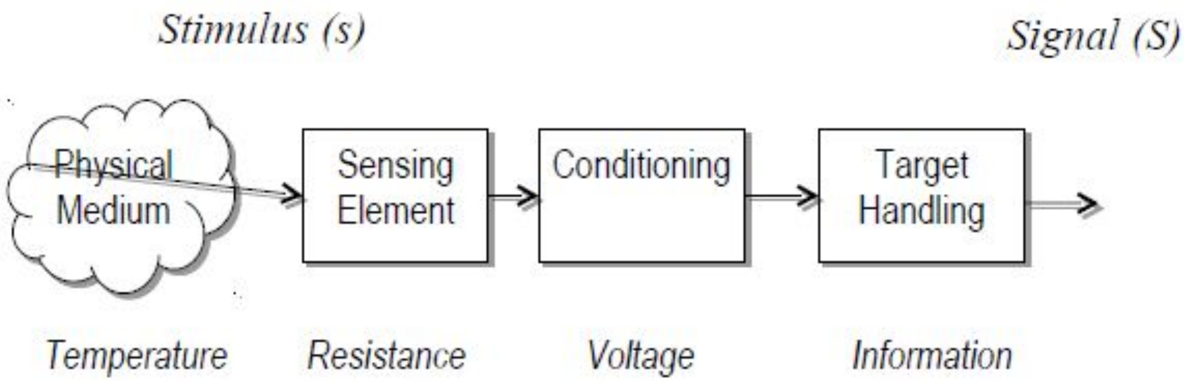
SENSORS

To implement process control, the computer must collect data from and transmit signals to the process

- Components required to implement the interface:
  - Sensors to measure continuous and discrete process variables
  - Actuators to drive continuous and discrete process parameters
  - Devices for ADC and DAC
  - I/O devices for discrete data



Control System



Transducers

Micro-sensors 10^{-6}m

Sensor

- Sensors are also called as detectors.
- The changes in the system environment or variables are detected by the sensors connected to the input port of the embedded system.
- It is a transducer that converts energy from one type to another type for any particular purpose.
- Example- ECG machine it is designed to monitor the heart beat status of a patient and it canot impose a control over the patient’s heart beat and its order. The sensors are used here are the different electrode sets connected to the body of the patient.
- The variations are captured and presented to the user through a visual display or some printed chart.

A sensor is a transducer that converts a physical stimulus from one form into a more useful form to measure the stimulus

Two basic categories:

1. Analog
2. Discrete

- Binary
- Digital (e.g., pulse counter)

Ultrasonic Sensor (distance)

Light Sensor (light intensity)

Sound Sensor (db pressure)

Touch Sensor

Other Sensors

- Temperature
- RFID
- Barcode
- Proximity
- Vision
- Gyroscope
- Compass
- Tilt/Acceleration
- Etc.

ACTUATORS

Hardware devices that convert a controller command signal into a change in a physical parameter

- The change is usually mechanical (e.g., position or velocity)
- An actuator is also a transducer because it changes one type of physical quantity into some alternative form
- An actuator is usually activated by a low-level command signal, so an amplifier may be required to provide sufficient power to drive the actuator

- Actuator is a form of transducer device which converts signals to corresponding physical action.
- Actuator acts as an output device.
- If the embedded system is designed for any controlling purpose the system will produce some changes in the controlling variable to bring the controlled variable to the desired value. This is achieved through an actuator connected to the output port of the embedded system.
- If the E.S is designed for monitoring purpose only then there is no need for including an actuator in the system.

Electrical actuators

- Electric motors

- DC servomotors
- AC motors
- Stepper motors

- Solenoids

1. Hydraulic actuators

- Use hydraulic fluid to amplify the controller command signal

1. Pneumatic actuators

- Use compressed air as the driving force

EMBEDDED FIRMWARE

- Embedded firmware refers to the control algorithm(program instructions) and configuration settings that an embedded system developer dumps into the code(program) memory of the embedded system.
- It is unavoidable part of embedded system
- There are different methods available to develop embedded firmware. Those are:

1. Write the program in high level languages like embedded C/C++ using an Integrated Design Environment.
2. Write the program in Assembly language using the instructions supported by your applications target processor/controller.

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"

- 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
- For a beginner in the embedded software field, it is strongly recommended to use the high level language based development technique.
- The reasons for this being writing codes in a high level language is easy, the code written in high level language is highly potable which means you can use the same code to run on different processor/controller with little or less modification.

System components

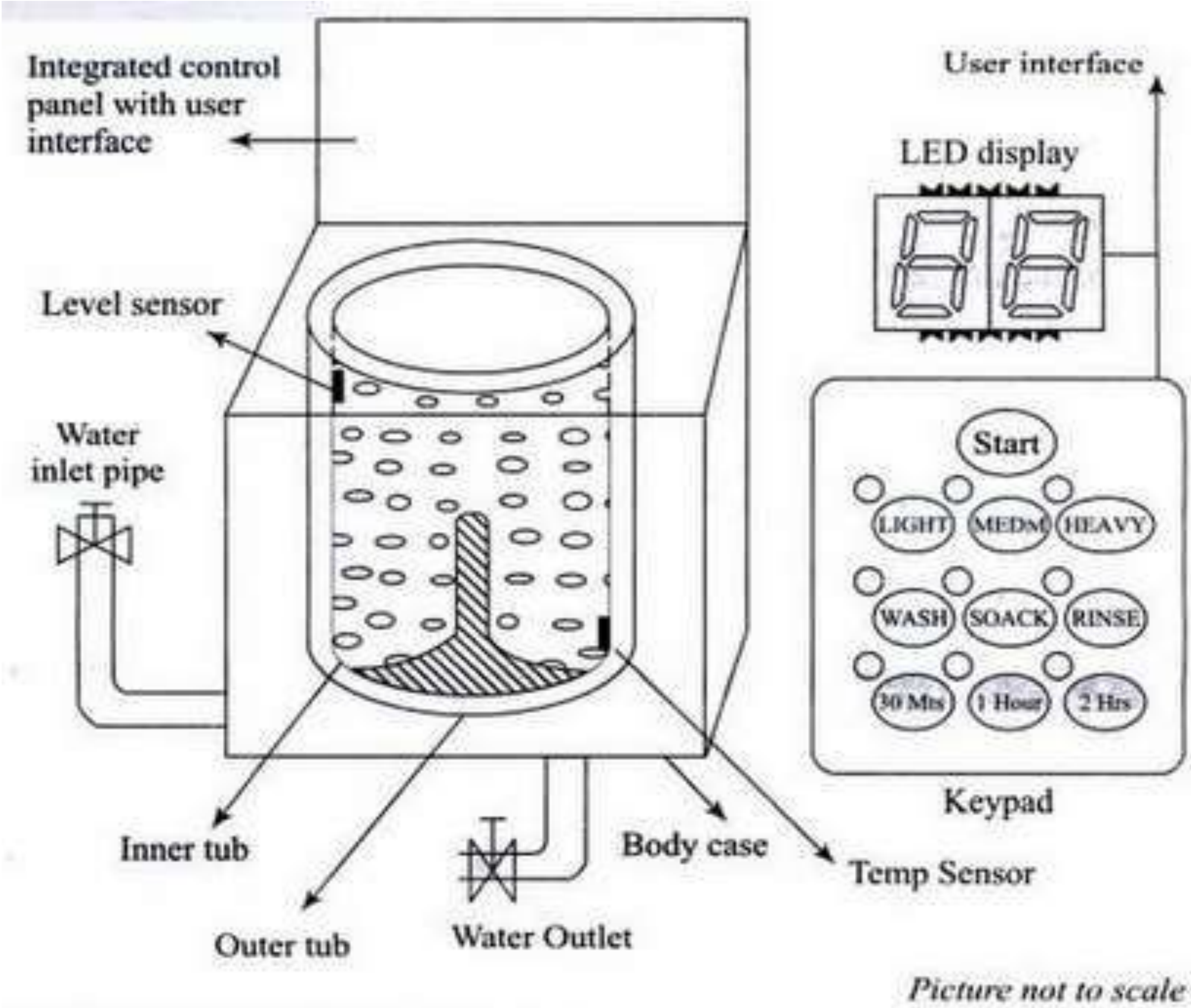
- Reset circuit
- Brown-out protection circuit
- Oscillator unit
- Real-time clock
- Watch dog timer

(**have to explain each and every term)**

Application - specific embedded system- **Washing Machine**

- Embedded systems are application specific means they are specifically built for certain applications

- Washing Machine is a typical example of an embedded system providing extensive support in home automation applications.
- An embedded system contains **sensors, actuators, control unit and application-specific user interfaces like keyboards, display units, etc.**
- You can see all these components in a washing machine if you have a closer look at it. Some of them are visible and some of them may be invisible to you.



The functional block diagram of a washing machine

- The **actuator part** of washing machine consists of a motorized agitator, tumble tub, water drawing pump and inlet valve to control the flow of water into the unit.
- The **sensor part** consists of the water temperature sensor, level sensor, etc.
- The **control part** contains a microprocessor/controller based board with interfaces to the sensors and actuators.
- The **sensor data** is fed back to the control unit and the control unit generates the necessary actuator outputs.
- The **control unit** also provides connectivity to user interfaces like keypad for setting the washing time, selecting the type of material to be washed like light, medium, heavy duty, etc.
- **User feedback** is reflected through the display unit and LEDs connected to the control board.

- The integrated control panel consists of a microprocessor/controller based board with I/O interfaces and a control algorithm running in it.
- Input interface includes the keyboard which consists of wash type selector namely *Wash, Spin and Rinse, cloth type selector namely Light, Medium, Heavy duty and washing time setting, etc.*
- The output interface consists of LED/LCD displays, status indication LEDs, etc. connected to the I/O bus of the controller.
- It is to be noted that this interface may vary from manufacturer to manufacturer and model to model.
- The other types of I/O interfaces which are invisible to the end user are different kinds of sensor interfaces, namely, water temperature sensor, water level sensor, etc. and actuator interface including motor control for agitator and tub movement control, inlet water flow control, etc.

Automotive-Domain-Specific Embedded System

- The major application domains of embedded systems are consumer, industrial, automotive, telecom, etc. of which telecom and automotive industry holds a big market share.

Inner Workings of Automotive Embedded Systems

- Automotive embedded systems are the one where electronics take control over the mechanical systems.
- The presence of automotive embedded system in a vehicle varies from simple mirror and wiper controls to complex air bag controller and antilock brake systems (ABS).
- Automotive embedded systems are normally built around microcontrollers or DSPs or a hybrid of the two and are generally known as Electronic Control Units (ECUs).

- The various types of electronic control units (ECUs) used in the automotive embedded industry can be broadly classified into two-High speed embedded control units and Low speed embedded control units.
- **High speed Electronic Control Units (HECUs)** : High speed electronic control units (HECUs) are deployed in critical control units requiring fast response, like fuel injection systems, antilock brake systems, etc.
- **Low speed Electronic Control Units (LECUs)** : Low speed electronic control units are deployed in applications where response time is not so critical. They are generally built around low cost microprocessors/microcontrollers and digital signal processors. Audio controllers, passenger and driver door locks, door glass controls, etc., are examples for LECUs.

Automotive Communication Buses

- Automotive applications use serial buses for communication.
- Controller Area Network (CAN), Local Interconnect Network (LIN), Media Oriented System Transport (MOST) bus, etc. are the important automotive communication buses.
- **CAN** is an event driven serial protocol interface with support for error handling in data transmission. It is generally employed in safety system like airbag control, power train systems like engine control and Antilock Brake Systems.

- **LIN** bus is a single master multiple slave (up to 16 independent slave nodes) communication interface. LIN is a low speed, single wire communication interface with support for data rates up to 20 kbps and is used for sensor/actuator interfacing.
- The **Media Oriented System Transport** (MOST) bus is targeted for automotive audio video equipment interfacing. MOST bus is a multimedia fiber-optic point-to-point network implemented in a star, ring or daisy chained topology over optical fibers cables.