SHORT NOTES

(Youtube)

Basics of Embedded Sy...

System → Arrangement in which all c... its follow ... of Rules. (assembled) (defined by s/w)
→ way of working

Embedded System — something that is attached to another thing
— Independent system / part of a large system.

Components — Hardware
— Software
— RTOS (Real Time Operating System)
→ supervise, monitors
→ set of Rules are defined to be followed
→ How a system should work.

Examples → Washing m/c, dish washer, microwave, Automobiles, walky talky, satellites, Smart watches, Air Bags, Antilock Braking system, Router, mobile phones, Aerospace – speed control, Temp. controller.

Advantage → Small size (Portable), low power consumption, low cost, durable ~~ratio~~ with high speed.

Features / characteristics of Embedded System

→ It is combination of H/w & S/w.
→ It is designed to perform a particular task.
→ The task has to be completed in a given time. (eg.– washing m/c).

INTRODUCTION TO EMBEDDED SYSTEMS

What is Embedded systems:-

→ An embedded system is a combination hardware and software designed for a speific function.

→ Embedded systems may also function within a larger system.

→ The systems can be programmable or have a fixed functionality.

Embedded means ??

→ Embedded system always function as part of a complete device → That's what meant by the term embedded.

→ They are low-cost, low power consuming, small computers that are embedded in other mechanical or electrical systems.

Generally Embedded systems comprised of

(1) processor (2) power supply (3) memory (4) communication ports.

Communication Port → transmits data between the processor & peripheral devices (often other embedded systems) using communication protocol.

Processor → interprets this data with the help of minimal software stored in memory.

Software → is usually highly speific to the function that the embedded system serves.

Examples → Industrial machines
consumer electronics
agricultural & processing industry devices
automobiles
medical equipment.
cameras,
digital watches
household appliances

Few more Definitions of Embedded System

(2)

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

OR

" A microprocessor based system that does not look like a computer".

OR

" A combination of computer Hardware & Software and perhaps additional mechanical or other parts, designed to perform a dedicated function. In some cases, embedded systems are part of a larger system or product, as is the case of an antilock braking system in a car".

OR

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

OR

(Formal definition from Book)

→ The word "Embedded processor" implies that a CPU is sharing the silicon wafer (bed) along with its program memory, a no. of output devices, and plenty of peripheral functions.

→ Thus an embedded processor is a self contained entity, which functions all by itself when powered.

→ Its intelligence depends only on the software present with ~~its~~ in its memory. It is thus provided with the built-in programmable peripherals.

⇒ SIGINIFICANCE

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

Definition (Tata McGraw Hill)

"An Embedded system is a system that has embedded software and computer hardware, which makes it a system dedicated for an application or specific part of an application or product or a part of a larger system".

→ Embedded systems are electronic systems that contain a microprocessor or microcontroller, but we do not think of them as computers - the computer is hidden or embedded in the system.

Examples of Embedded Systems :-

Embedded systems are used in a wide range of Technologies across an array of industries Some Examples Include :-

1) Automobile

→ modern cars commonly consist of many computers or embedded systems, designed to perform different tasks within the vehicle. Some of these systems perform basic utility functions and other provide entertainment or user facing functions. Some embedded systems in consumer vehicles include cruise control, backup sensors, suspension control, navigation systems and airbag systems.

2) Mobile Phones :-

These consist of many embedded systems, including GUI software and hardware, operating systems (OSes) cameras, microphones, and USB I/O modules.

3) Industrial Machines :

They can contain embedded systems, like sensors, and can be embedded systems themselves. Industrial machines often have embedded automation systems that perform specific monitoring and control functions

4) Medical Equipment :

Eg. for eg. sensors & control mechanisms

Applications of the embedded system in Various Areas.

1. Telecom
   - mobile Computing
   - mobile Access.
2. Smart Cards
   - Banking.
   - Security
3. Missiles
   - Defence
   - Aerospace
   - Communication
4. computer Networking systems + Peripherals.
   - Networking Systems
   - Image processing
   - Printers
   - Networks Cards
   - monitors + Displays.
5. Digital Consumer electronics
   - DVDs.
   - set top boxes.
   - High Definition TVs.
   - Digital Cameras.
6. Automotive
   - motor Control System.
   - Cruise Control
   - Engine / Body safety.
   - Robotics in Assembly Line
   - Car entertainment
   - Car multimedia.

Examples of Embedded Systems

Examples of Small scale Embedded System

1. Automatic choclate vending machine.
2. Stepper/motor controllers for a Robotics system.
3. Washing or cooking systems.
4. Multitasking Toys.
5. CD drive / hard disk drive controller.
6. Peripheral controllers of a computer
   Eg:- CRT display controller, keyboard controller.
   DRAM controller, printer controller,
   Laser printer controller, LAN controller.
   disk drive controller.
7. Fax / photocopy / printer / scanner m/c.
8. Remote of TV.
9. Telephone with memory, display & other sophisticated features.
10. Motor control systems → an accurate control of speed & position of the d.c motor, Robot & CNC m/c, Automotive Applications such as closed loop engine control, dynamic ride control, Antilock braking systems monitor
11. electronic instruments, such as industrial process controller.
12. Electronic smart weight display system.
13. Spectrum Analyzer.

Examples of Medium Scale Embedded System

1. computer networking system → Router, front end processor in a server, a switch, a bridge, a hub & a gateway.
2. Video game & music system.
3. Banking system, (Bank ATM, credit card transaction).
4. Signal tracking system, (automatic signal tracker.)
5. mobile communication SIM card / cellular phone / cable TV / Fax transceiver
6. Image filtering, Image processing, speech processing, video processing.

Examples of Sophisticated Embedded system

1. mobile smart phones & computing system.
2. mobile computer.
3. Embedded systems for wireless LAN.
4. Embedded systems for video, Real time video and speech or multimedia processing systems.
5. security products and high speed network security.

COMPONENTS OF EMBEDDED SYSTEM HARDWARE

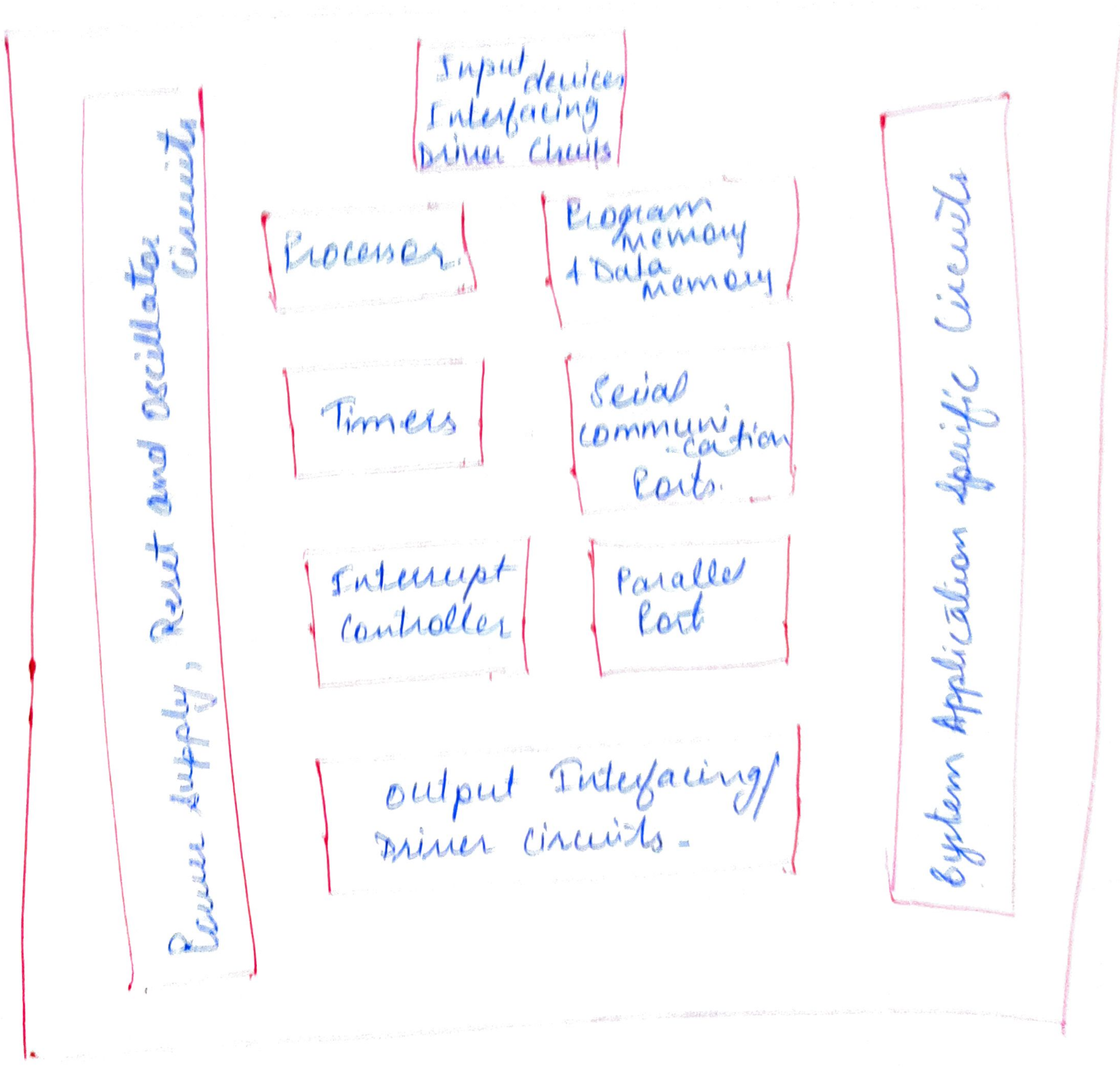


Fig 1.1 → The components of Embedded System Hardware

Components of Embedded Systems

① It embeds hardware similar to a computer. As its software usually embeds in the ROM or flash memory, it usually do not need a secondary hard disk and CD memory as in a computer.

② It embeds main application software. The application software may concurrently perform a series of tasks or processes of or threads.

③ It embeds a Real-Time Operating System :→ that supervises the application software running on hardware and organizes access to a resource according to the priorities of tasks in the system. It provides a mechanism to let the processor run a process as scheduled and context switch between the various processes. A small scale embedded system may not embed the RTOS.

Characteristics of Embedded Systems:-

(1) Real Time and Multirate operations define the ways in which the system works, reacts to events, interrupts and schedules the system's functioning in Real Time.

→ It does so by following a plan to control latencies and to meet deadlines

(Latency refers to the waiting period between running the codes of a task or interrupt service routine and the instance at which the need for the task or interrupt from an event arises).

The different operations may take place at distinct rates. For Eg.→ Audio, video, data, network stream & events have diff. rates and time constraints

(2) Complex Algorithms.

(3) Complex Graphic User Interface (GUIs) and other user interfaces.

(4) Dedicated functions.

Constraints :-

An embedded system is designed keeping in view 3 constraints

① Available system- memory
② Available processor speed.
③ the need to limit power dissipation when running the system continuously in cycles of wait for events 'Run', 'stop', 'wake-up' and 'sleep'.

Embedded Processors in a System

→ A processor is an important unit in the embedded system hardware.

→ It is heart of the embedded system.

→ knowledge of basic concept of microprocessors and microcontrollers is must for an embedded system designer.

→ A processor has 2 essentials units → Flow control unit (FCU)
→ Execution unit (EU)

→ FCU → includes a fetch unit for fetching instructions from the memory.

EU → has circuits to implement the instructions pertaining to data transfer operations and data conversions from one form to another

→ It includes ALU (Arithmetic & Logical unit) and also the circuits that execute instructions for a program control task → say, Halt, Interrupt, Jump to another set of instructions

→ A processor runs the cycles of fetch-and-execute. The instructions, defined in the processor instruction set, are executed in the sequence that they are fetched from the memory. A processor is in the form of an IC chip

Design Process in Embedded System (1.8 Tata Mc graw)

The concepts used during a design process are as follows.

1. **Abstraction** :- Each problem component is first abstracted. For eg— in the design of a Robotic system, the problem of abstraction can be in terms of control of arms and motors.

2. **Hardware & Software Architecture** : Architectures should be well understood before a design.

3. **Extra Functional Properties** :- Extra functionalities required in the system being developed should be well understood from the design.

4. **Modular Design** : → modular design concepts should be used. system designing is fast by decomposition of software into modules that are to be implemented.
→ modules should be clearly understood and should maintain continuity.
→ Also, appropriate protection strategies are necessary for each module. A module is not permitted to change or modify another module functionality.
for eg:- protection from a device driver modifying the configuration of another device.

6. **Mapping** : Mapping into various representation is done from software requirements.

For Eg. data flow in the same path during the program flow can be mapped together as a single entity. Transform and transaction mapping design processes are used in designing.

For Eg. → an image is input data to a system; it can have a different no. of pixels & colours. The system does not process each pixel & colour individually. mapping of image is done by Transform appropriate compression and storage algorithms. Transaction mapping is done to define the sequence of images.

7. **User Interface Design** :

→ It is important part of design.

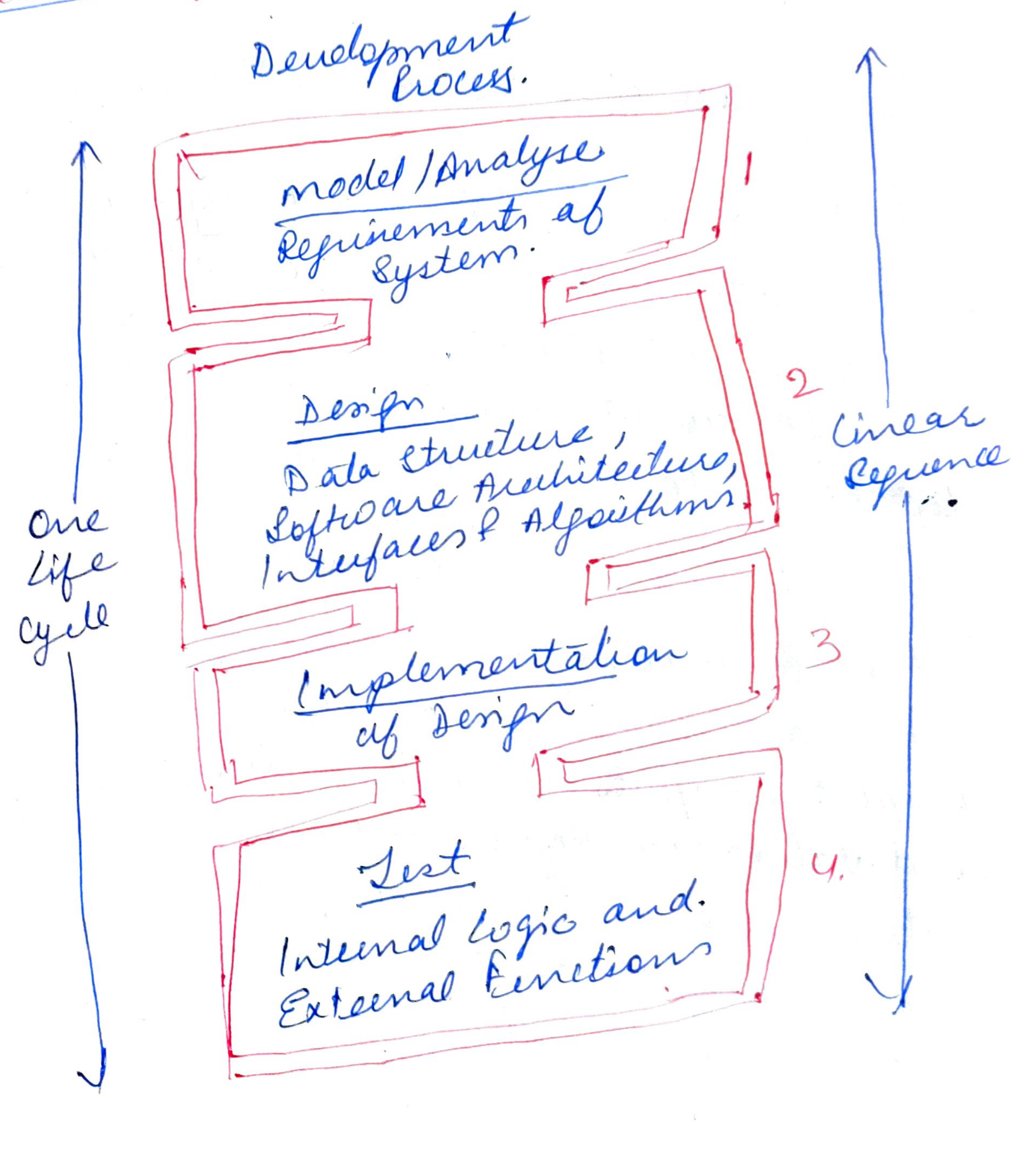
→ User interfaces are designed as per user requirements, analysis of the environment and system functions.

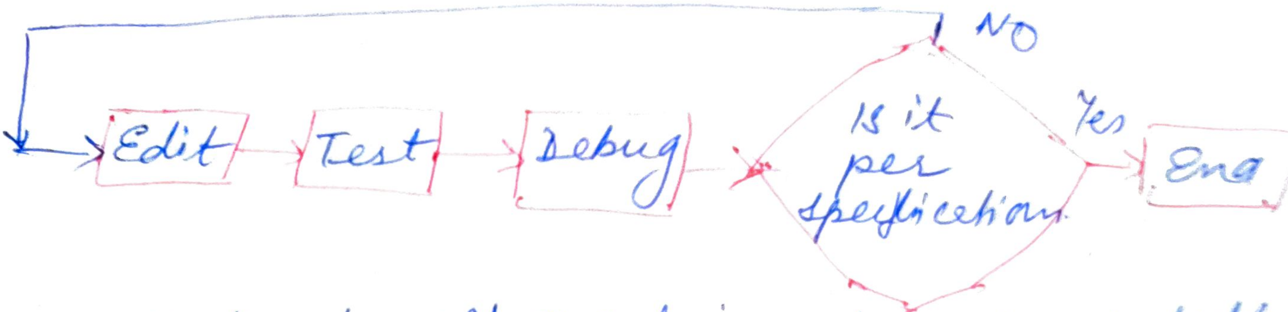
For eg. In an automatic chocolate vending machine (ACVM) system, the user interface is an LCD multiline graphics display. It can display a welcome message as well as specify the coins needed to be inserted into the machine for each type of chocolate.

→ The same ACVM may be designed with Touchscreen User Interface (GUI), or it may be designed with Voice User Interfaces (VUIs).

(8) **Refinements** :— Each component & module design needs to be refined iteratively till it becomes the most appropriate for implementation by the software team.

The activities for software-design cycle during an embedded software development process.





Eg: Activities for software design during an embedded software-development process

Abstraction of steps in the Design Process

→ A design process is called Top-to-bottom if it first starts with abstraction of the process and then after abstraction the details are created.

→ The following lists the 5 levels of abstraction from top of bottom in the design process.

① Requirements :— Definition and analysis of system Requirement

→ It is only by a complete clarity of the required purpose, inputs, outputs, functioning, design metrics and validation Requirements for finally developed systems specifications that a well designed system can be created. There has to be consistency in the Requirements

2 Specifications :

→ clear specifications of the required system are must.
→ specifications need to be precise.
→ specifications are required for.
(i) data Hardware — for eg. — peripherals, devices processor and memory specifications.

(ii) data-types & processing specifications
(iii) Expected system behaviour specifications.
(iv) constraints of design.
(v) Expected life cycle specifications.

3 Architecture : — data modelling designs of attributes of data structure, data flow graphs, program model, software architecture layers and hardware architecture are defined.

Software Architectural layers

Ist layer is → Architectural Design → design for a system architecture is developed → Questions arised → How the diff. elements — data structures, databases, algorithms, control functions, state transition functions, process, data & program flow — are to be organised.

IInd layer → data design → Questions → 1 what design of data structures & databases would be most appropriate for the given problem.? 2 whether data organised as a tree like structure will be appropriate? 3 what will be the design of the components in the data. [For eg:— video information will have 2 component — Image and Sound].

III^rd layer → Interface Design → Questions → (1) What shall be the interfaces to integrate the components.

(2) What is the design for system integration.

(3) What shall be design of interfaces used for taking inputs from the data objects, structures and databases and for delivering outputs

(4) What will be the port structure for receiving inputs and transmitting outputs?.

(4) COMPONENTS :

→ Fourth layer is a component level design.

→ Question at this stage → what shall be the design of each component? ~~There is an~~ following list the common hardware components :

1. Processor, ASIP & single purpose processors in the system.
2. Memory RAM, ROM or Internal & external flash or secondary memory in the system.
3. Peripherals and devices internal & external to the system.
4. Ports & Buses in the system.
5. Power source or Battery in the system.

(5) System Integration → Built components are integrated in the system.

Components may work fine independently but when integrated may not fulfill the design metrics.

Challenges in Embedded System Design

1. Amount and Type of Hardware needed :

Optimising the requirement of microprocessor, ASIP (Application Specific Instruction Set Processor) and Single purpose processors in the system on the basis of performance, power dissipation, cost and other design metrics are the challenges in a system design.

→ A designer also chooses the appropriate hardware (memory RAM, ROM or internal & external flash or secondary memory, peripheral 4 devices internal + External ports and buses and power source or battery) taking into account the design metrics.

2. Optimising power Dissipation and Consumption:

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

Methods used to meet the Design Challenges

1. Clock Rate Reduction :—
2. Voltage Reduction.
3. Wait, Stop and Cache Disable Instructions :—

DESIGN PROCESS AND DESIGN EXAMPLES

Automatic Chocolate Vending Machine (ACVM)

Let us consider an automatic chocolate vending machine. This interesting example given here helps a reader to understand several concepts of programming an embedded system as a multitasking system.

Figure 1.12 shows the diagrammatic representation of ACVM. Assume that ACVM has following components:

1. It has keypad on the top of the machine. That enables a child to interact with it when buying a chocolate. The owner can also command and interact with the machine.
2. It has an LCD display unit on the top of the machine. It displays menus, text entered into the ACVM and pictograms, welcome, thank you and other messages. It enables the child as well as the ACVM owner to graphically interact with the machine. It also displays time and date. (For GUIs, the keypad and LCD display units or touch screen are basic units.)
3. It has a coin insertion slot and a mechanical coin sorter so that child can insert coins to buy a chocolate.
4. It has a delivery slot so that child can collect the chocolate and coins, if refunded.
5. It has an Internet connection port using a USB based wireless modem so that owner can know status of the ACVM sales from a remote location.

ACVM Functions Assume that ACVM functions are as follows:

1. The ACVM displays the GUIs and if the child wishes to enter contact information, birthday information or get answer to FAQs, it displays the appropriate menu.
2. It displays a welcome message when in idle state. It also continuously displays time and date at the right bottom corner of display screen. It can also intermittently display news, weather data or advertisements or important information of interest during idle state.
3. When first coin is inserted, a timer also starts. The child is expected to insert all required coins in 2 minutes.
4. After 2 minutes the ACVM will display a query to the child if the child does not insert sufficient coins. If the query is not answered the coins are refunded.
5. Within 2 minutes if sufficient coins are collected, it displays the message, 'Thanks, wait for few moments please!', delivers the chocolate through the delivery slot and displays message. Collect the chocolate and visit again, please!'

Hardware units: ACVM embeds the following hardware units.

1. Microcontroller or ASIP(Application Specific Instruction Set Processor)
2. RAM for storing temporary variables and stack
3. ROM for application codes and RTOS codes for scheduling the tasks
4. Flash memory for storing user preferences, contact data, user address, user date of birth, user identification code, answers of FAQs
5. Timer and interrupt controller
6. A TCP/IP port (Internet broadband connection) to the ACVM for remote control and for lhe owner to get ACVM status reports

7. ACVM specific hardware to sort coins of different denominations. Each denomination coin generates a set of status and input bits and port-interrupts. Using an ISR forthat port, the ACVM processor reads the port status and input bits. The bits give the information about which coin has been inserted. After each read operation, the status bits are reset by the routine
8. Power supply

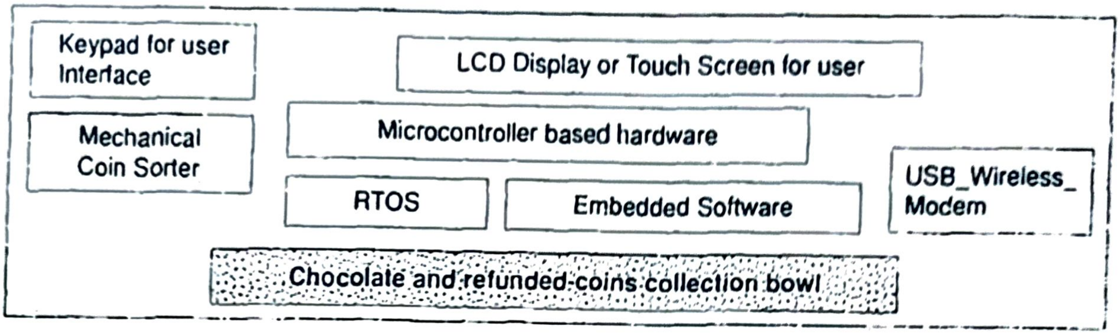


Fig. 1.12 Diagrammatic representation of the ACVM

Software components: ACVM embeds the following software components:
1. Keypad input read task
2. Display task
3. Read coins task for finding coins sorted
4. Deliver chocolate task
5. TCP/IP stack processing task
6. TCP/IP stack communication task

Smart Card

Smart card is one of the most used embedded system today. It is used for credit-debit bankcard, ATM card, e-purse or e-Wallet card, identification card, medical card (for history and diagnosis details) and card for a number of new innovative applications.
The smart card is a plastic card ISO standard dimensions, 85.60 x 53.98 x 0.80 mm. It is an embedded system on a card: SoC (System-On-Chip). ISO recommended standards are ISO78I6 (1 to 4) for host-machine contact-based cards and ISO14443 (Part A or B) for the contactless cards. The silicon chip is just a few multimeters in size and is concealed in-between the layers. Its very small size protects the card from bending. Figure 1.13 shows embedded-system hardware components for a contactless smart card.

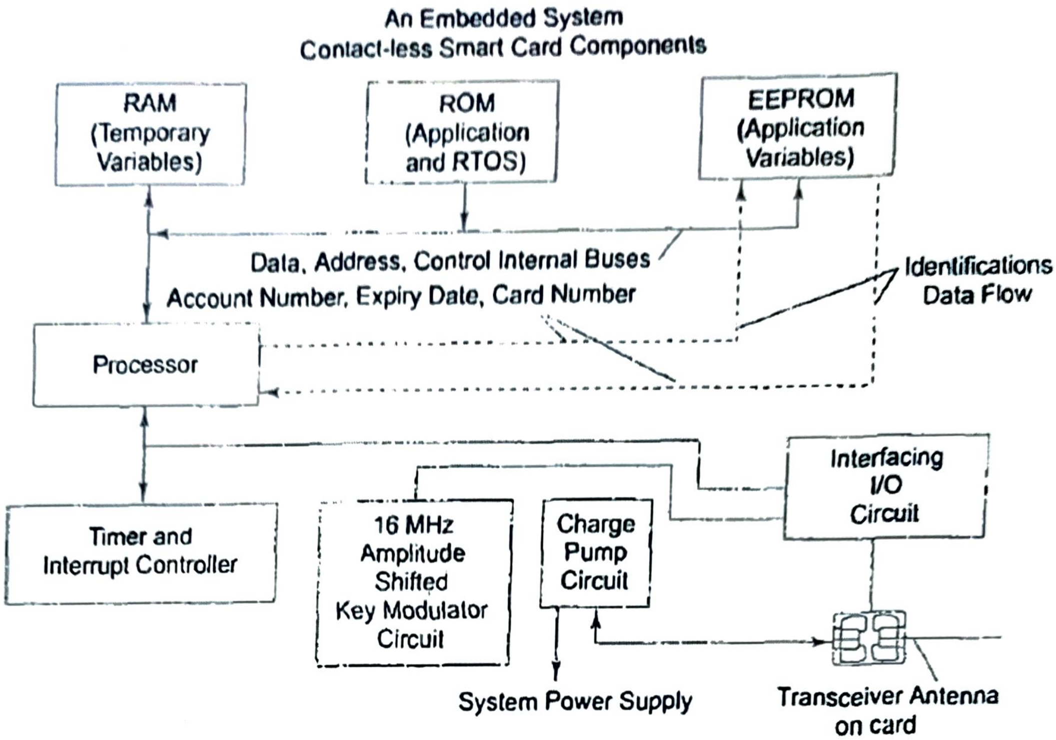


Fig. 1.13 Embedded hardware components in a contact less smart card

Embedded Hardware The embedded hardware components are as follows:

- Microcontroller or ASIP
- RAM for temporary variables and stack
- One time programmable ROM for application codes and RTOS codes for scheduling the tasks
- Flash for storing user data, user address, user identification codes, card number and expiry date
- Timer and interrupt controller
- A carrier frequency -16 MHz generating circuit and Amplitude Shifted Key (ASK) modulator
- Interfacing circuit for the IOs
- Charge pump for delivering power to the antenna for transmission and for system circuits. The charge pump stores charge from received RF (radio frequency) at the card antenna in its vicinity. [The charge pump is a simple circuit that consists of the diode and high value ferroelectrics material-based capacitor.

Embedded Software Smart card embeds the following software components:

- Boot-up, initialisation and OS programs
- Smart card secure file system
- Connection establishment and termination
- Communication with host
- Cryptography algorithm
- Host authentication
- Card authentication
- Saving addition parameters or recent new data sent by the host (for example, present balance left)

Mobile Phone

- The mobile phone today has a large number of features. It has sophisticated hardware and software.

Hardware units A mobile phone embeds an SoC (System-on-Chip) integrating lhe following hardware units.

1. Microcontroller or AS1P |An ASIP is configured to process encoding and deciphering and another does the voice compression. Third ASIC dials, modulates, demodulates, interlaces the keyboard and touch screen or multiple line LCD graphic displays, and processes the data input and recall of data from memory |.
2. DSP core, CCDSP, DSP, video, voice and pixel processors
3. Flash, memory stick, EEPROMs and SRAMs
4. Peripheral circuits, ADC, DAC and interrupt controller
5. Direct memory access controller (Section 4.8)
6. LCD controller (Section 3.3.4)
7. Battery

What is the Internet of Things (IoT)?

The Internet of Things, or IoT, is a network of physical devices. These devices can transfer data to one another without human intervention.

IoT devices are not limited to computers or machinery. The Internet of Things can include anything with a sensor assigned a unique identifier (UID).

The primary goal of the Internet of Things is to create self-reporting devices that can communicate with each other (and users) in real time.

In India, the use of the IoT is quickly increasing as the implementation of 5G networks expands across the country. As the use of IoT continues to grow, the need for professionals with knowledge of IoT and how it works is likely to increase in demand.

How does IoT work?

Here are a few components that make the Internet of Things work:

- **An Internet of Things platform:** An IoT platform manages device connectivity. It can be a software suite or a cloud service. The purpose of an IoT platform is to manage and monitor hardware, software, processing abilities, and application layers.
- **Sensor technologies:** IoT sensors, sometimes called smart sensors, convert real-world variables into data that devices can interpret and share. Many different types of sensors exist. For example, temperature sensors detect heat and convert temperature changes into data. Motion sensors detect movement by monitoring ultrasonic waves and triggering a desired action when those waves are interrupted.
- **Unique identifiers:** The core concept of the IoT is communication between devices and users. Unique identifiers (UIDs) establish the context of a device within the more extensive network to enable this communication. Identifiers are patterns, like numeric or alphanumeric strings. One example of a UID that you might be familiar with is an internet protocol (IP) address. They can identify a single device (instance identifier) or the class to which that device belongs (type identifier).
- **Connectivity:** Sensors can connect to cloud platforms and other devices through a host of Internet network protocols.
- **Artificial intelligence (AI) and machine learning:** Natural language processing (NLP) in IoT devices makes it easier for users to input information and interact with them. Machine learning also enhances their analytical capabilities.

IoT applications

Billions of devices are connected to the internet, collecting and sharing information. They range from smart home setups like cooking appliances and smoke detectors to military-grade surveillance equipment. The list below outlines a few of the most common types of IoT applications.

- **Consumer IoT:** Consumer IoT refers to personal and wearable devices that connect to the internet. These devices are often referred to as smart devices.

- **Industrial Internet of Things (IIoT):** The IIoT is a system of interconnected devices in the industrial sector. Manufacturing machinery and devices used for energy management are part of it.
- **Commercial IoT:** Commercial IoT refers to tools and systems used outside of the home. Businesses and health care organisations leverage commercial IoT for auditable data trails and consumer management.

Internet of Things examples

As mentioned above, the Internet of Things has many different applications. You likely use IoT devices every day. Some that you may be familiar with include smart thermostats and kitchen appliances, fitness-tracking watches, self-driving cars, and home security systems. Personal medical devices like pacemakers are also IoT devices.

Real Life Example of Working of IoT
In our data to day life, we use many IoT based devices. Some of them are:

- Nowadays, many people wear smartwatches which is none other than an IoT device. It contains an accelerometer that measures the number of steps taken, detects hand movements, etc.
- Using (GPS), these devices can determine your location and compute the distances traveled.
- There are many IoT devices which are now making your home smart. You can lock/unlock you doors using an application. There are many devices designed which has sensors that can detect any type of mishappening in your home, eg: Glass break, smoke, heat, motion detectors, etc.
- Not only this, IoT devices can also help in disaster management. They help in detection of temperature, carbon content in the region. In case of any forest fire, these devices send an alert to the control room, fire department.

There are many more IoT devices that we are using in our daily life like smartphones, smart cars, biometric attendance, face detection lock/unlock, etc.

Major Components of IoT (Internet of Things)

There are 5 major components of IoT (Internet of Things) - Devices or Sensors, Gateway, Cloud, Analytics and User Interface:

1. Sensors or Devices

Sensors or Devices are basically used to collect and transmit the data and also perform actions based on those data. For example, the sensors can be used for measuring temperature and humidity. There are different types of sensors; here are as follows: Temperature Sensors, Humidity Sensors, Proximity Sensors, Motion Sensors, Light Sensors, Pressure Sensors, Gas Sensors, and GPS Sensors

2. Gateway

Gateway is also a device component that basically acts as an intermediate between the sensors and the central cloud. Gateway is one of the essential components of IoT that offers communication, management, and data processing. Here are some of the functions of Gateway in IoT: Data Aggregation, Communication, Security, Protocol Translation, Load Balancing, and Latency Reduction.

3. Cloud

Cloud in IoT refers to the service that provides the management, storage, and processing of the data that is generated by IoT (Internet of Things) devices. Here are some key aspects of Cloud in IoT: Data Storage, Data Collection, Security, Connectivity, Integration, and Cost Efficiency.

4. Analytics

This is the crucial component of IoT that basically harness the potential of IoT. In analytics, meaningful insights are analyzed that are generated by IoT devices and sensors. There are some functions included in Analytics, such as data processing, machine learning, and statistical analysis. Here are some of the applications of analytics in IoT: Anomaly Detection, Environmental Monitoring, Energy Management, Smart Cities, and Agriculture.

5. User Interface

User Interface, also known as UI in the Internet of Things (IoT) and provides an interface by which the users can interact with the applications and systems. Here are some of the key points in the user interface of IoT (Internet of Things): Data Visualization, User-Friendly Design, Personalization, Remote Management, Integration, Authentication, and Security.

IOT Design Methodology — Steps

1. **Purpose & Requirements** → Define purpose & Requirements of IOT system.
2. **Process model Specification**: Define the use cases.
3. **Domain model Specification**: Define Physical entities, virtual entities, Devices, Resources and Services in IOT system.
4. **Information Model Specification**: define the structure (eg.- Relations, attributes) of all info^n in IOT system.
5. **Service Specifications**: — Map Process and Information model to services and define service specifications.
6. **IOT Level Specification**: Define the IOT level for the system
7. **Functional View Specification**: Map IOT level to functional groups.
8. **Operational View Specification**: Define communication options, service hosting options, storage options, device options.
9. **Device & Component Integration**: Integrate devices, develop and integrate the components.
10. **Application Development**: Develop Application.

Step 1

→ A use case is a concept in Software engineering that describes how a system is used to achieve a goal or task. It's way to identify, clarify & organize system requirements by out-lining the interactions b/w a system & users.

Step 1 : Purpose & Requirements Specification

→ The first step in IOT system design methodology is to define the purpose and requirements of the system. In this step, the system purpose, behavior and requirements (such as data collection requirements, data analysis requirements, system management requirements, data privacy and security requirements, user interface requirements) are captured.

Step 2 : Process Specification:

→ The second step in the IOT design methodology is to define the process specification. In this step, the use cases of the IOT system are formally described based on and derived from the purpose & requirement specifications.

Step 3 : Domain Model Specification :

→ The third step in the IOT design methodology is to define the Domain model. The domain model describes the main concepts, entities and objects in the domain of IOT system to be designed.

→ Domain model defines the attributes of the objects and relationships between objects.

→ Domain model provides an abstract representation of the concepts, objects & entities in the IOT domain, independent of any specific technology or platform.

~~Arduino~~

~~→ open source electronics platform.~~

~~→ can control from a small bulb to power grid~~

~~→ used to design or build prototypes of electronic projects.~~

→ With the domains, independent of any specific technology or platform. With the domain model, the IOT system designers can get an understanding of the IOT domain for which the system is to be designed.

→ Domain model defines the attributes.

→ With the domain, independent of any speific technology or platform. With the domain model, the IOT system designers can get an understanding of the IOT domain for which the system is to be designed.

→ Domain model defines the attibutes of the objects and relationships between objects and entities in the IOT domain, independent of any speific technology or platform.

→ With the domain model, the IOT system designers can get an understanding of the IOT domain for which the system is to be designed.

Step4 Information Model Speification :-

→ The fourth step in the IOT design methodology is to define the Information model,

→ Information model defines the structure of all the Information in the IOT system, for example, attributes of Virtual Entities, relations, etc. Information model does not describe the speifics of how the information is represented or stored.

Step 5 Service Specifications

→ Service Speifications define the services in the IOT system, service Types, Service I/Ps / O/Ps, Service endpoints, Service Schedules, Service Preconditions and service effects.

Step 6 IOT level Specification

→ IOT level-1 system has a single node/device that performs sensing and/or actuation, stores data, performs analysis & hosts the application

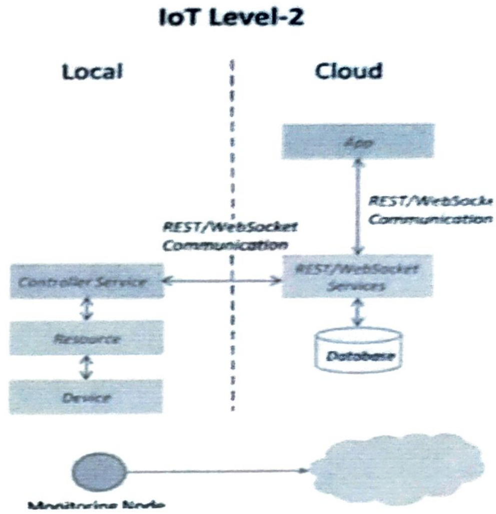
→ This level is suitable for modelling low cost and low complexity solutions where the data involved is not big and analysis requirements are not computationally intensive

Eg: Home Automation System.

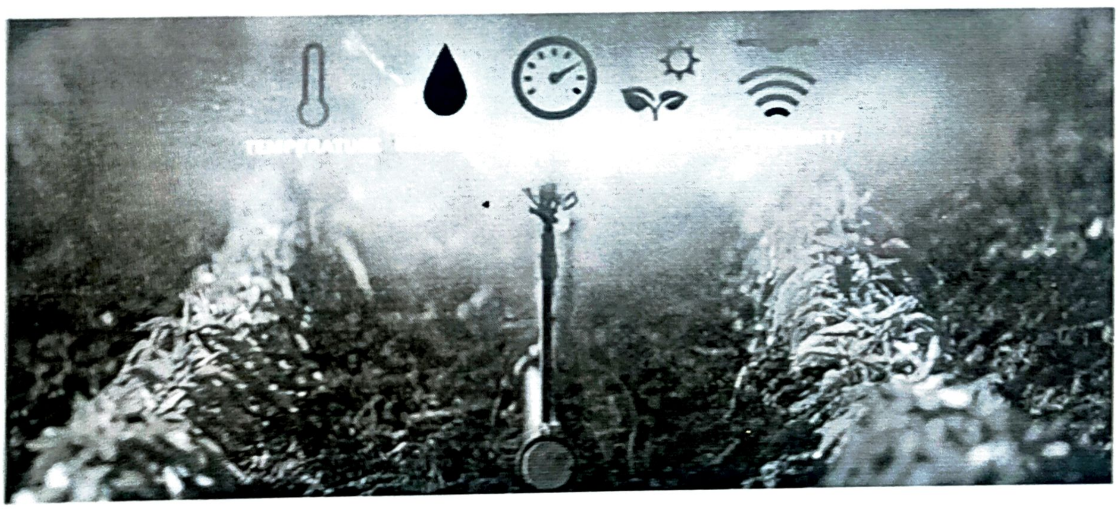
IoT Level-2

- A level-2 IoT system has a **single node** that performs sensing and/or actuation and **local analysis**. **Data is stored in the cloud** and the **application is usually cloud-based.**

- Level-2 IoT systems are suitable for solutions where the data involved is big; however, the primary analysis requirement is not computationally intensive and can be done locally.



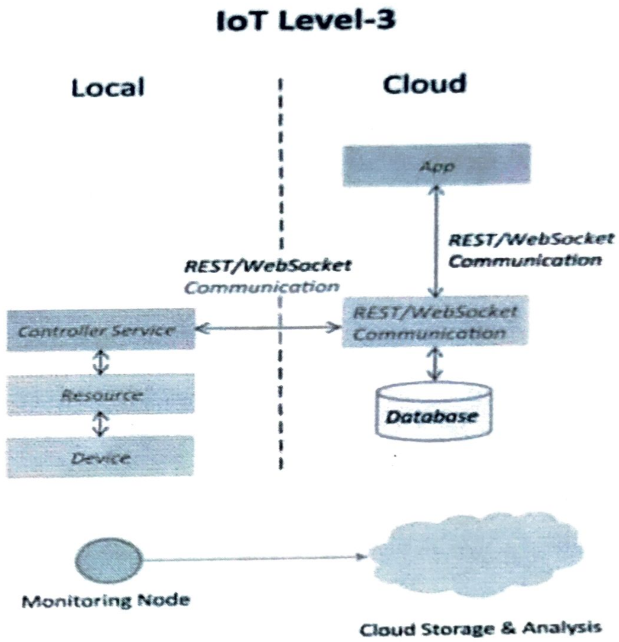
IoT – Level 2 Example: Smart Irrigation



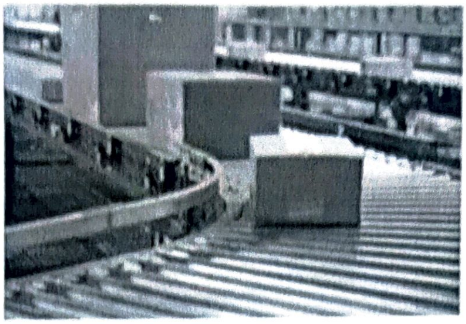
IoT Level-3

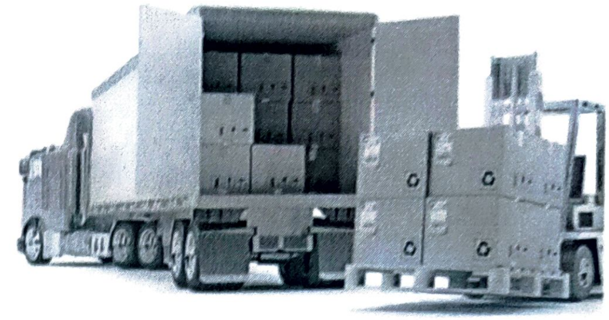
A level-3 IoT system has a **single node**. Data is stored and **analyzed in the cloud** and the **application is cloud-based.**

Level-3 IoT systems are suitable for solutions where the **data involved is big** and the **analysis requirements are computationally intensive**.



IoT – Level 3 Example: Tracking Package Handling

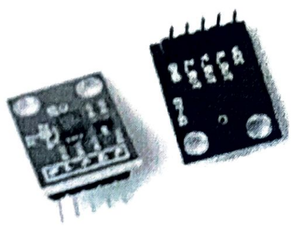




Sensors used

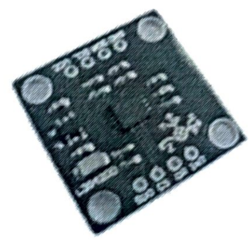
Accelrometer

sense movement or vibrations



Gyroscope

Gives orientation info



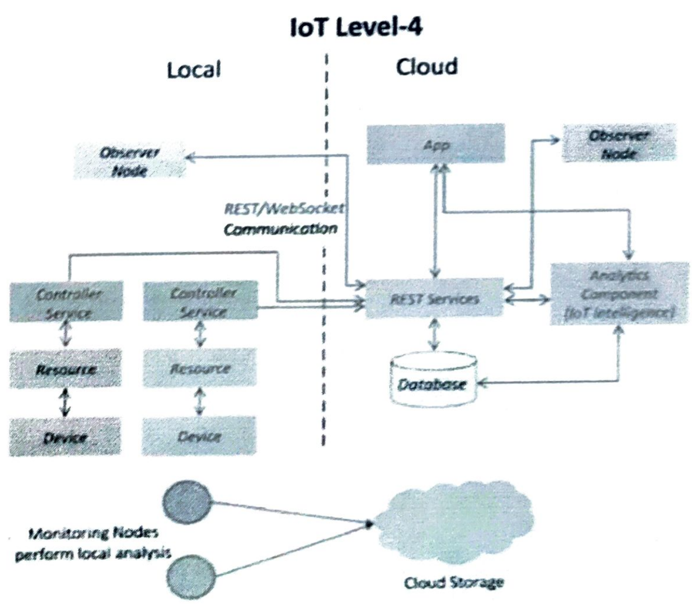
Websocket service is used because sensor data can be sent in real time.

IoT Level-4

A level-4 IoT system has **multiple nodes** that perform **local analysis. Data is stored in the cloud** and the application is cloud-based.

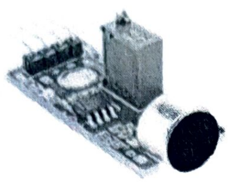
Level-4 contains local and cloud-based **observer nodes** which can subscribe and receive information collected in the cloud from IoT devices.

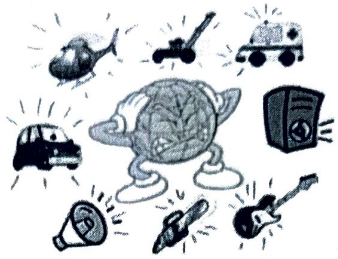
Level-4 IoT systems are suitable for solutions where **multiple nodes are required, the data involved is big and the analysis requirements are computationally intensive.**

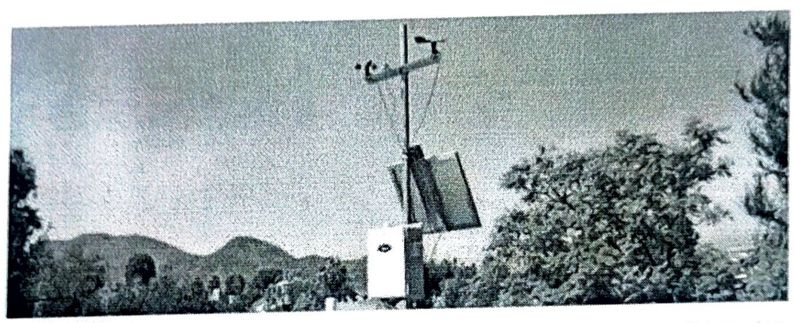


IoT – Level 4 Example: Noise Monitoring

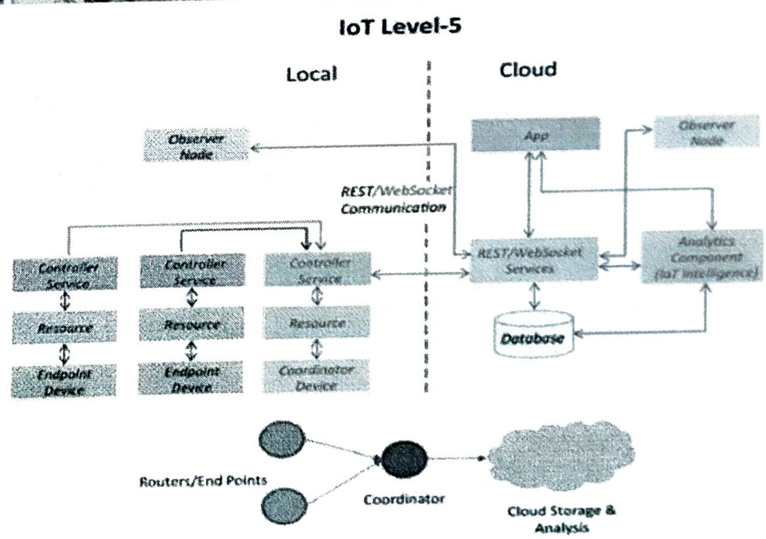
Sound Sensors are used







IoT Level-5



- A level-5 IoT system has **multiple end nodes** and **one coordinator node.**
- The end nodes perform sensing and/or actuation.
- The coordinator node **collects data from the end nodes and sends it to the cloud.**
- Data is stored and **analyzed in the cloud** and the **application is cloud- based.**

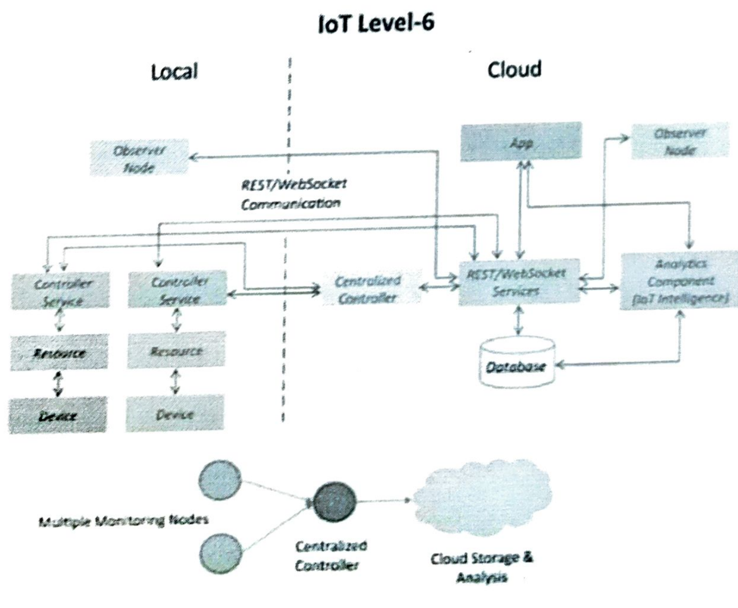
Level-5 IoT systems are suitable for **solutions based on wireless sensor networks**, in which the **data involved is big** and the **analysis requirements are computationally intensive.**

IoT – Level 5 Example: Forest Fire Detection

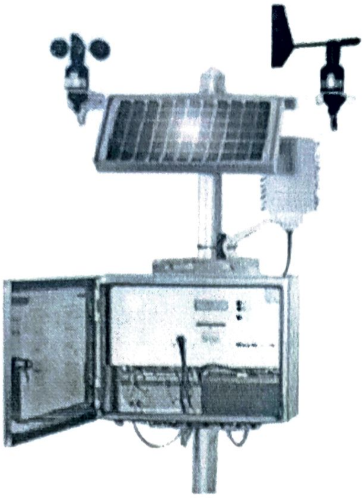
Detect forest fire in early stages to take action while the fire is still controllable.
Sensors measure the temperature, smoke, weather, slope of the earth, wind speed, speed of fire spread, flame length

IoT Level-6

•A level-6 IoT system has **multiple independent end nodes** that perform sensing and/or actuation and send data to the cloud.
•**Data is stored in the cloud and the application is cloud-based.**
•The **analytics** component analyzes the data and stores the results in the **cloud database.**
•The results are visualized with the cloud-based application.
•The **centralized controller** is aware of the status of all the end nodes and **sends control commands to the nodes.**



IoT – Level 6 Example: Weather Monitoring System



Sensors used

Wind speed and direction
Solar radiation
Temperature (air, water, soil)
Relative humidity

Precipitation
Snow depth
Barometric pressure
Soil moisture

IoT Issues and Challenges

Security
- Cyber Attacks, Data Theft

Privacy
- Controlling access and ownership of data.

InterOperability
- Integration Inflexibility

Legality and Rights
- Data Protection laws be followed, Data Retention and destruction policies

Economy and Development
- Investment Incentives, Technical Skill Requirement

Step 7: Functional View Specification

- The seventh step in the IoT design methodology is to define the Functional View. The Functional View (FV) defines the functions of the IoT systems grouped into various Functional Groups (FGs). Each Functional Group either provides functionalities for interacting with instances of concepts defined in the Domain Model or provides information related to these concepts.

Step 8: Operational View Specification

- The eighth step in the IoT design methodology is to define the Operational View Specifications. In this step, various options pertaining to the IoT system deployment and operation are defined, such as, service hosting options, storage options, device options, application hosting options, etc

Step 9: Device & Component Integration

- The ninth step in the IoT design methodology is the integration of the devices and components.

Step 10: Application Development

- The final step in the IoT design methodology is to develop the IoT application.