# Embedded Systems Architecture-Review

**Prachi Prakash Jagtap**

*B.E, Department of Electrical Engineering, AISSMS College of Engineering, Pune, Maharashtra, India*

**Abstract** – *The most broadly used systems in Electrical, electronic, computer fields are Embedded Systems. A system is nothing but arrangement in which all its unit assemble work together according to a set of rules. It can also be defined as a way of working, organizing or doing one or many tasks according to a fixed plan. For example, a watch, the speedometer, remote controller, computers etc. To understand the embedded systems, its vey imperative o get familiar with its architecture and working. An embedded gadget can be a notion of as a laptop hardware having a software embedded in it. An embedded gadget may be an impartial system, or it may be part of a huge system. An embedded system is a microcontroller or microprocessor primarily based gadget that's designed to perform a task. For instance, a fire alarm is an embedded system; it'll sense most effective smoke.*

***Key Words***: **Embedded system, operating system, software, hardware, architecture, application, memory, signal, processor**

## 1. INTRODUCTION

An embedded system is a programmed controlling and operating system with a dedicated function within a larger mechanical or electrical system, often with real-time computing constraints. It is embedded as part of a complete device often including hardware and mechanical parts.

Modern embedded systems are often based on microcontrollers (i.e. CPUs with integrated memory or peripheral interfaces), but ordinary microprocessors (using external chips for memory and peripheral interface circuits) are also common, especially in more-complex systems. In either case, the processor(s) used may be types ranging from general purpose to those specialized in certain class of computations, or even custom designed for the application at hand.

### 1.1 Components of embedded system architecture

An embedded system has 3 components:

It has the hardware.

It has software program.

It has an actual real-time operating system (RTOS) that supervises the utility software and offer a mechanism to let the processor run a process as in step with scheduling by means of following a plan to manipulate the latencies. RTOS defines the manner the system works. It units the rules throughout the execution of application software. A small scale embedded device won't have RTOS

### 1.2 The embedded system model

An expansion of embedded systems architectural structures is used to introduce technical concepts and fundamentals of an embedded device. the emerging architectural equipment (i.e., reference models) had been used as the inspiration for these architectural systems.at the best degree, the primary architectural tool used to introduce the important factors located inside an embedded device layout is what I can consult with as the embedded systems model, shown in below figure.

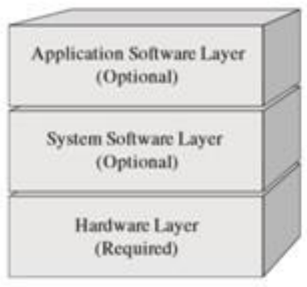
Application Software Layer (Optional)
System Software Layer (Optional)
Hardware Layer (Required)

**Fig -1**: Embedded system model

## 2. BASIC STRUCTURE OF EMBEDDED SYSTEMS ARCHITECTURE

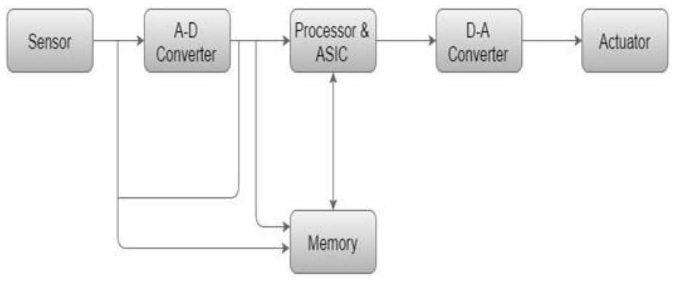

**Fig -2**: Components of embedded system

SENSOR: It measures the quantities that are physical and converts it to an electrical signal which may be read by an observer or through any electronic tool like an A-D converter. A sensor shops the measured amount to the memory.

A-D CONVERTER: An analog-to-digital converter that is used converts the analog signal sent by using the sensor right into a digital signal.

PROCESSOR: Processors process the records to degree the output and keep it to the memory.

D-A CONVERTER: A virtual-to-analog converter converts the virtual records fed by using the processor to analog information.

ACTUATOR: An actuator compares the output given by means of the D-A converter to the actual (anticipated) output saved in it and stores the authorized output.

## 2.1 Characteristics of embedded systems

**Single-functioned** – An embedded system usually performs a specialized operation and does the same repeatedly. For example: A pager always functions as a pager.

**Tightly constrained** – All computing systems have constraints on design metrics, but those on an embedded system can be especially tight. Design metrics is a measure of an implementation's features such as its cost, size, power, and performance. It must be of a size to fit on a single chip, must perform fast enough to process data in real time and consume minimum power to extend battery life.

**Reactive and Real time** – Many embedded systems must continually react to changes in the system's environment and must compute certain results in real time without any delay. Consider an example of a car cruise controller; it continually monitors and reacts to speed and brake sensors. It must compute acceleration or de-accelerations repeatedly within a limited time; a delayed computation can result in failure to control of the car.

**Microprocessors based** – It must be microprocessor or microcontroller based.

**Memory** – It must have a memory, as its software usually embeds in ROM. It does not need any secondary memories in the computer.

**Connected** – It must have connected peripherals to connect input and output devices.

**HW-SW systems -** Software is used for more features and flexibility. Hardware is used for performance and security.

## 3. IMPORTANCE OF EMBEDDED SYSTEM ARCHITECTURE

Making use of architectural structures engineering method to embedded systems due to the fact it's far one of the maximum powerful gear that can be used to recognize an embedded structures layout or to clear up demanding situations faced while designing a new device. What makes the architectural technique so effective is its capacity to informally and quick speak a layout to a spread of people with or without technical backgrounds, even acting as a basis in planning the assignment or certainly designing a device. Because it truly outlines the requirements of the system, an architecture can act as a solid basis for studying and testing the quality of a device and its performance below various situations. Eventually, the diverse systems of an architecture can then be leveraged for designing destiny merchandise with comparable traits, as a result allowing design understanding to be reused, and leading to a decrease of destiny design and development charges.

## 4. TYPES OF ARCHITECTURE

The 8051 microcontrollers work with 8-bit data bus. So they can support external data memory up to 64K and external program memory of 64k at best. Collectively, 8051 microcontrollers can address 128k of external memory.

When data and code lie in different memory blocks, then the architecture is referred as Harvard architecture. In case data and code lie in the same memory block, then the architecture is referred as Von Neumann architecture.

### 4.1 Von Neumann Architecture:

The Von Neumann architecture was first proposed by a computer scientist John von Neumann. In this architecture, one data path or bus exists for both instruction and data. As a result, the CPU does one operation at a time. It either fetches an instruction from memory or performs read/write operation on data. So, an instruction fetch and a data operation cannot occur simultaneously, sharing a common bus.

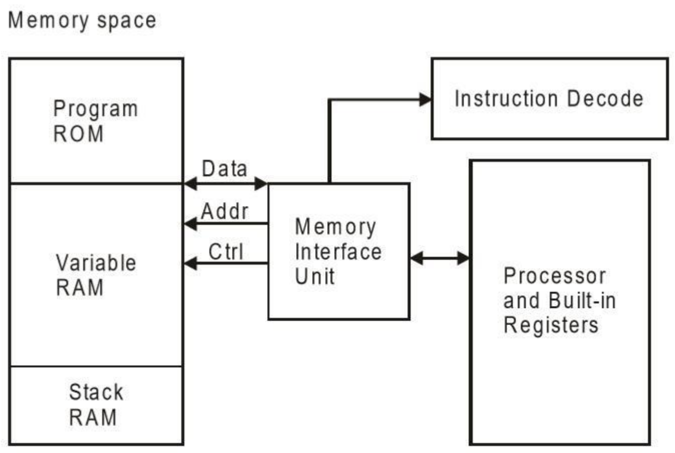

**Fig -3**: Von Neumann architecture

### 4.2 Harvard Architecture

The Harvard architecture offers separate storage and signal buses for instructions and data. This architecture has data storage entirely contained within the CPU, and there is no access to the instruction storage as data.

Computers have separate memory areas for program instructions and data using internal data buses, allowing simultaneous access to both instructions and data.

Programs needed to be loaded by an operator; the processor could not boot itself. In a Harvard architecture, there is no need to make the two memories share properties.

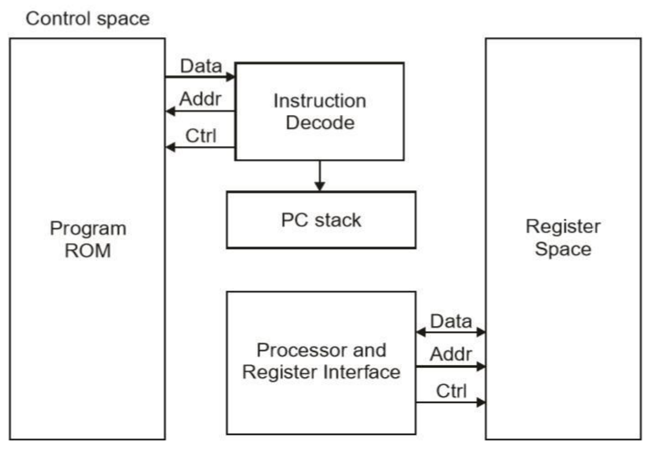

**Fig -4**: Harvard architecture

4.3 CISC AND RISC

CISC is a Complex Instruction Set Computer. It is a computer that can address many instructions. In the early 1980s, computer designers recommended that computers should use fewer instructions with simple constructs so that they can be executed much faster within the CPU without having to use memory. Such computers are classified as Reduced Instruction Set Computer or RISC.

## 5. Advantages and disadvantages of embedded systems

Advantages:

- Easily Customizable
- Low power consumption
- Low cost
- Enhanced performance
- It has fast operation.
- It has improved product quality.
- It optimizes use of system resources.

Disadvantages:

- High development effort
- The embedded systems are hard for maintenance as it is use and throw device.
- Less power supply durability if it is battery operated.
- It has hard to take backup of embedded files.

## 6. CONCLUSION

Since the embedded system is dedicated to specific tasks, design engineers can optimize it to reduce the size and cost of the product and increase the reliability and performance. Some embedded systems are mass-produced, benefiting from economies of scale. Embedded systems range from portable devices such as digital watches and MP3 players, to large stationary installations like traffic lights, factory controllers, and largely complex systems like hybrid vehicles, MRI, and avionics. Complexity varies from low, with a single microcontroller chip, to very high with multiple units, peripherals and networks mounted inside a large chassis or enclosure.

## REFERENCES

[1] J. Cook and J. Freudenberg, Embedded Software Architecture. EECS University of Michigan, 2007. [Online]. http://www.eecs.umich.edu/courses/eecs461/lecture/SWArchitecture.pdf

[2] A. Monot, M. Oriol, C. Schneider, and M. Wahler, “Modern software architecture for embedded real-time devices: High value, little overhead,” in 2016 13th Working IEEE/IFIP Conference on Software Architecture (WICSA), April 2016, pp. 201–210.

[3] I. Crnkovic, Building Reliable Component-Based Software Systems, M. Larsson, Ed. Norwood, MA, USA: Artech House, Inc., 2002.

[4] ——, “Component-based software engineering for embedded systems,” in Proceedings of the 27th International Conference on Software Engineering, ser. ICSE ’05. New York, NY, USA: ACM, 2005, pp. 712–713. [Online].