

A Presentation on

**Demystifying Embedded Systems: Components,
Applications, and Design**

Presented by :

Dr. Arun Kumar Singh
Dept. of Electronics Engineering
Rajkiya Engineering College, Kannauj

Verifying Embedded Systems: Components, Applications, Design

OUTLINE

- Introduction to Embedded System
- Embedded vs General Purpose Systems
- Basic Block Diagram
- Core Components of Embedded System
- Concept of Microprocessor & Microcontroller
- Types of Processing Architectures
- Instruction Set Architectures
- Taxonomy of Computer Architecture
- Hardware-Software Co-Design
- Communication Protocols
- Design Constraints
- Applications
- Emerging Trends

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

INTRODUCTION

An **embedded system** is a specialized computing system designed to perform a **specific task** within a larger system. It integrates **hardware** and **software** to operate efficiently under **real-time constraints**.

Unlike general-purpose computers, embedded systems are **optimized for performance, cost, and power**.

They are **everywhere in daily life**, from household devices to advanced industrial systems.

Examples:

Washing machine controller

Braking (ABS) system

Smartwatch / fitness band

Microwave oven

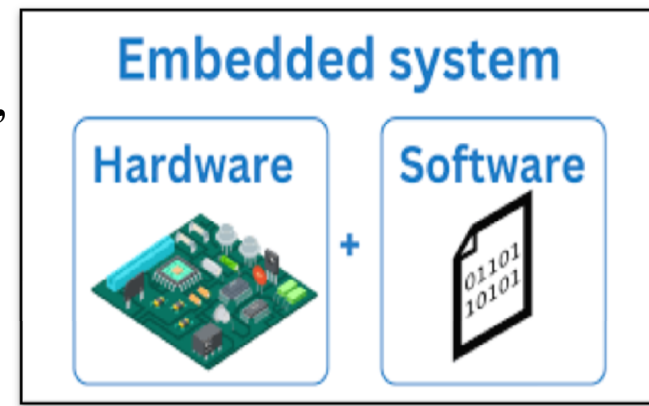


Fig 1(a). Embedded System Model

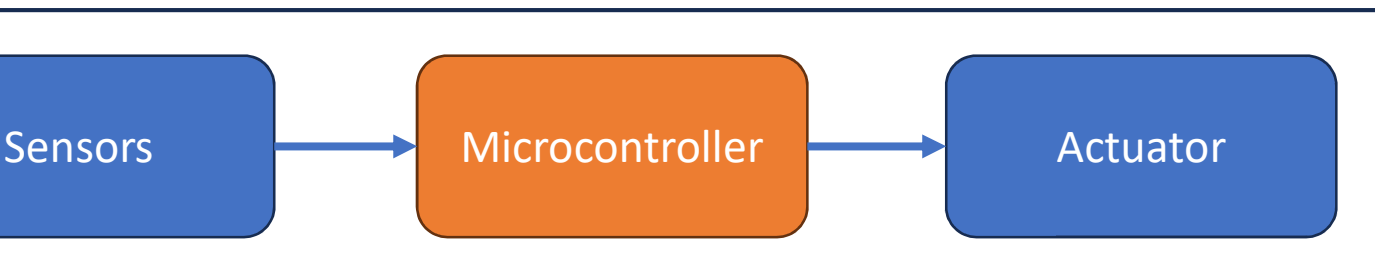


Fig 1(b). Embedded System Components

Composition of an Embedded System

Sensors: Detects physical inputs.

Microcontroller: Brain of the system.

Actuators: Convert signals into physical actions.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

EMBEDDED vs GENERAL PURPOSE SYSTEMS

	Embedded System	General-Purpose System
Task	Specific task	Multiple tasks
Flexibility	Limited	High
Performance	Optimized for one function	General performance
Interface	Minimal / none	Full interface
Examples	Washing machine, car ECU	Laptop, desktop

Fig 2(a). Embedded and General-Purpose system comparison

Embedded systems prioritize efficiency and reliability, while general-purpose systems prioritize flexibility.

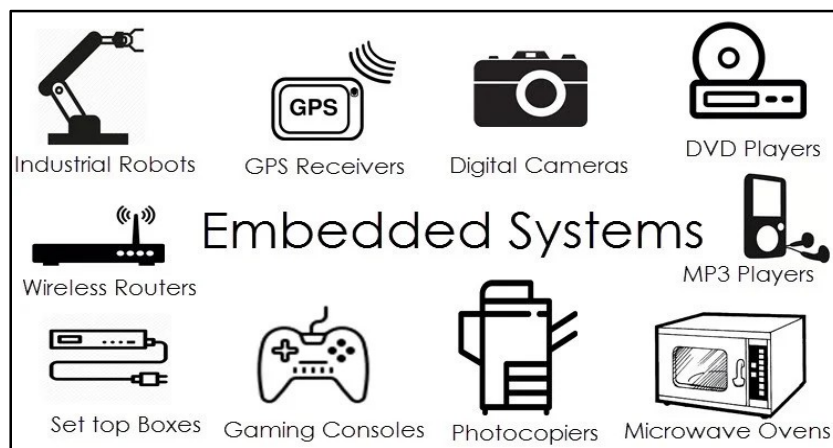


Fig 2(b). Examples of Embedded System



Fig 2(c). Examples of General-Purpose System

The key difference is purpose. Embedded systems are designed for a **dedicated task**, while general-purpose systems like laptops can handle **multiple applications**. This specialization makes embedded systems more efficient and cost-effective.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Block Diagram

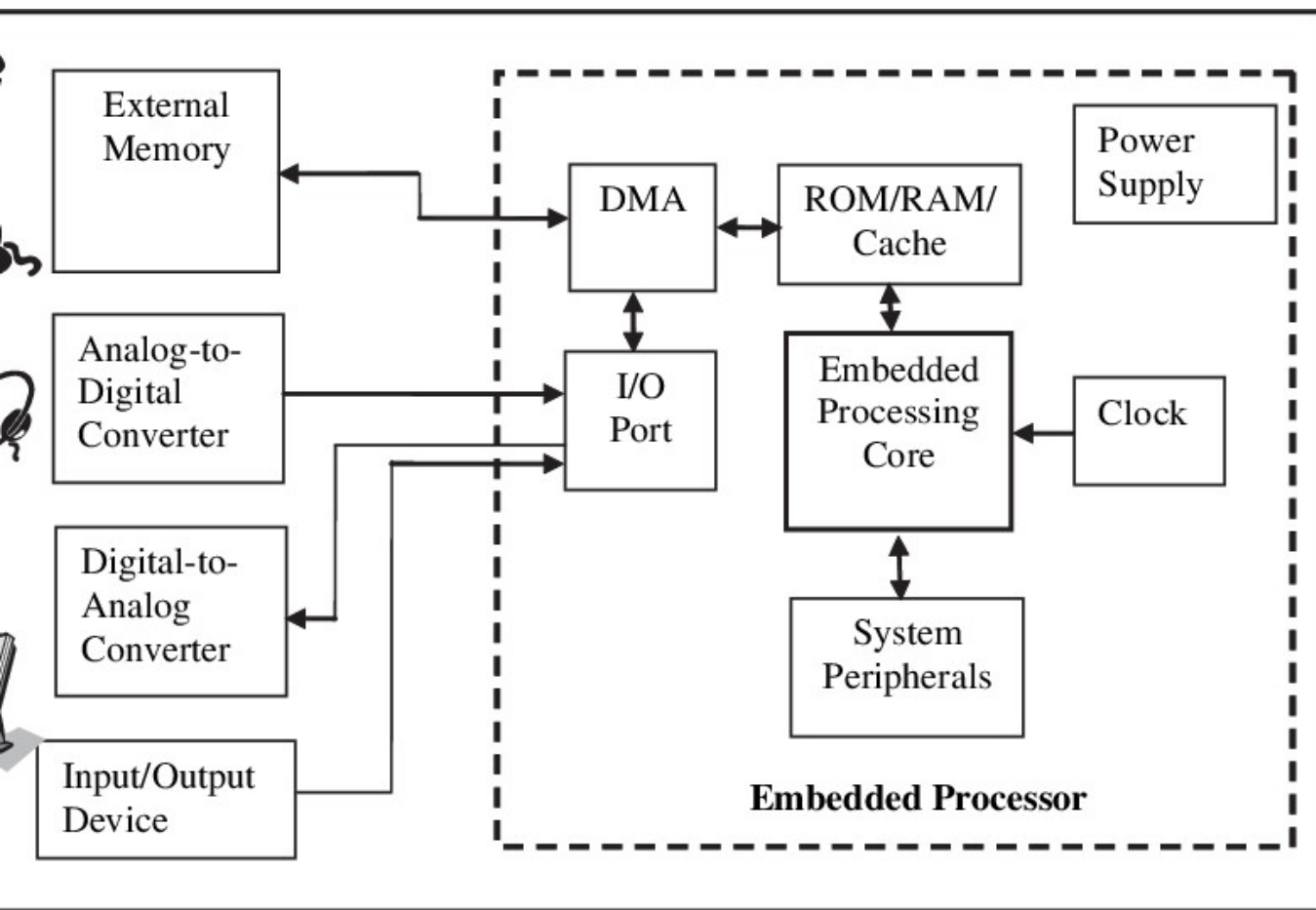


Fig 3(a). Block Diagram of Embedded System

Every embedded system follows this simple

Sensors collect data from the environment.
Microcontroller processes it based on programmed logic,
Actuators perform the required action.
Memory supports this process by storing instructions and data.

Modern embedded systems optimize performance by using **DMA** and cache memory to reduce load and latency.

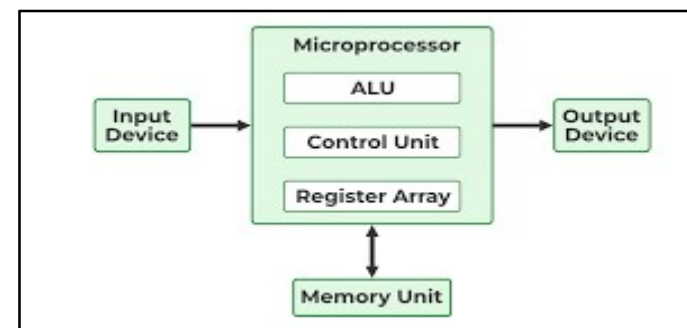


Fig 3(b). Simplified Block Diagram of Embedded System

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Components of Embedded Systems

Input Subsystem (Sensors + ADC):

Converts analog/physical signals and converts them into digital form for processing.

Processing Core (MCU/MPU):

Executes firmware, handles control logic, interrupt servicing, and real-time decision making.

Output Subsystem:

Flash/ROM: Program storage

RAM: Runtime data

Cache (if present): Reduces memory latency

Output Subsystem (DAC + Actuators):

Converts processed digital signals into physical actions.

Timing System: Defines execution timing and synchronization.

Timers/Counters: Used for scheduling, PWM generation, and event

Interfacing & Data Movement

I/O Interfaces: GPIO, UART, SPI, I2C, CAN.

DMA (Direct Memory Access): Enables high-speed data transfer without CPU intervention.

Interrupt Mechanism: Ensures low-latency response to external/internal events.

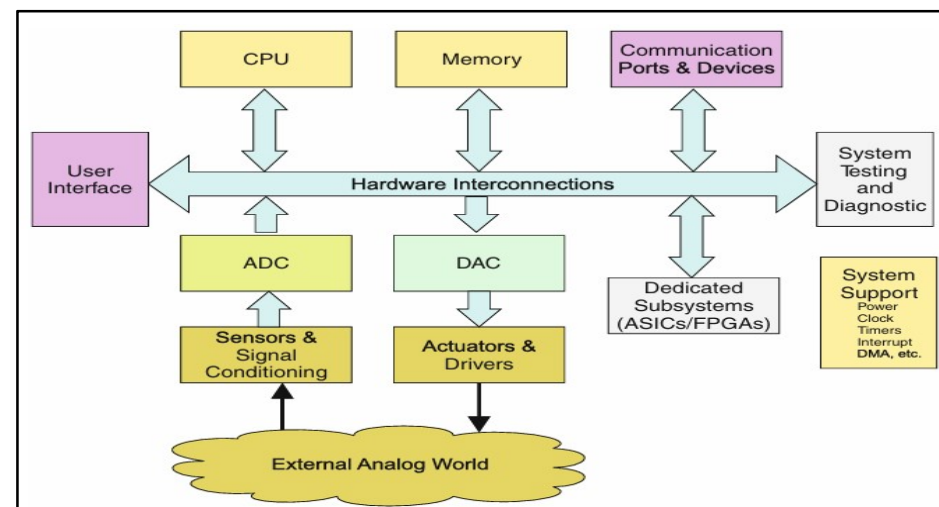


Fig 4(a). Components and its interfacing.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Comparison of Microprocessor & Microcontroller

Microprocessor (MPU)

A **general-purpose CPU** that requires **external memory and peripherals**.

Executes instructions fetched from external memory.

Typically used in **high-performance systems**.

Key Points:

Separate RAM, ROM, I/O

Higher computational capability

Runs full operating systems (Linux, Windows)

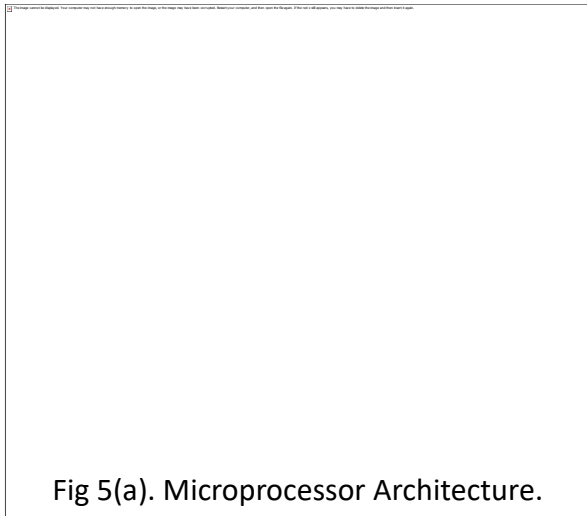


Fig 5(a). Microprocessor Architecture.

Microcontroller (MCU)

- A **self-contained system-on-chip (SoC)** integrating:
 - CPU + Memory + I/O + Peripherals.

- Designed for **dedicated embedded applications**.

Key Points:

- On-chip Flash, RAM, GPIO, timers

- Low power, low cost

- Deterministic behavior (real-time capable)



Fig 5(b). Microcontroller Architecture.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Types of Processing Architecture (Based on Memory Organization)

Von Neumann Architecture

Single memory for **instructions and data**
Shared bus → **Von Neumann bottleneck**
Simpler design, lower cost

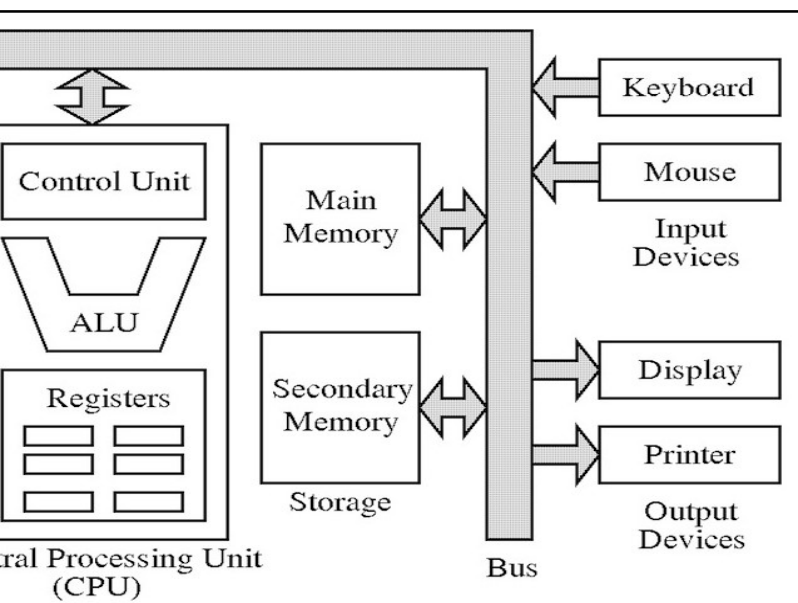


Fig 6(a). Von Neumann Architecture.

Harvard Architecture

- Separate memory for **instructions and data**
- Parallel access → **higher throughput**
- Widely used in embedded systems

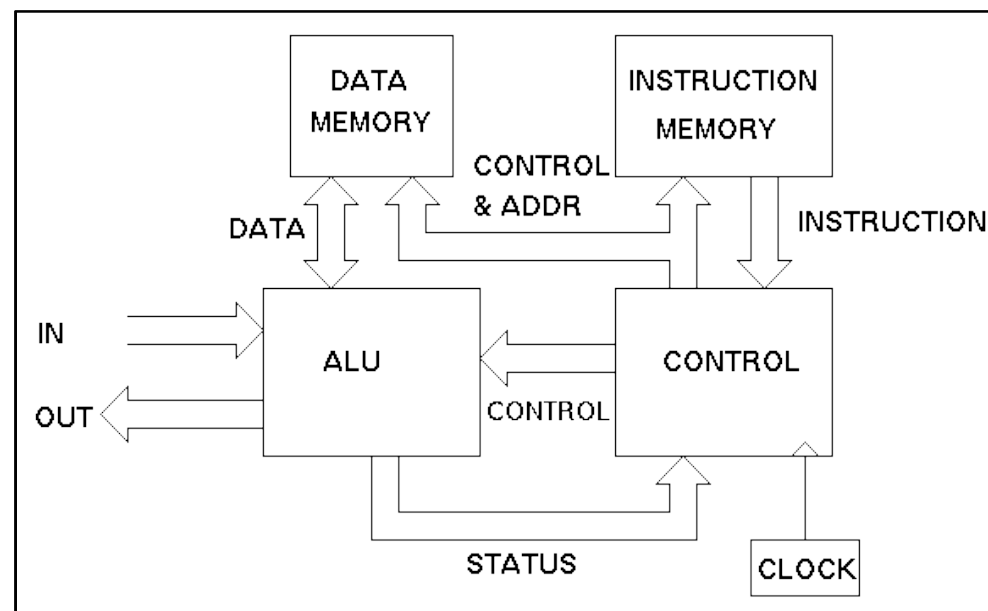


Fig 6(b). Harvard Architecture.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Types of Processing Architecture (Instruction Set Architecture)

CISC (Complex Instruction Set Computer)

Complex instructions, fewer cycles per task

Higher hardware complexity

Example: x86

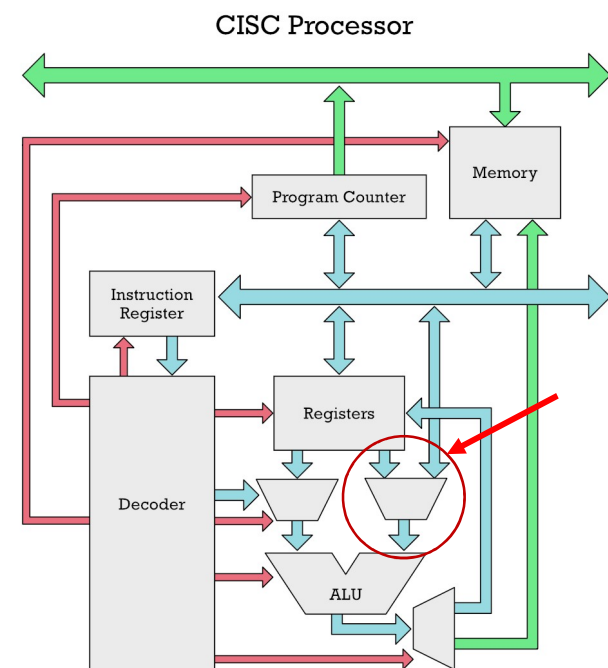


Fig 7(a). CISC Architecture.

RISC (Reduced Instruction Set Computer)

- Simple, fixed-length instructions
- Pipeline-friendly, faster execution
- Example: ARM, RISC-V

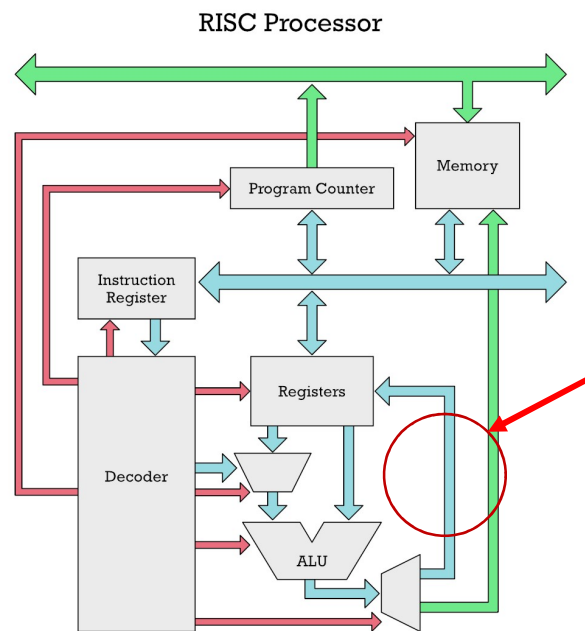


Fig 7(b). RISC Architecture.

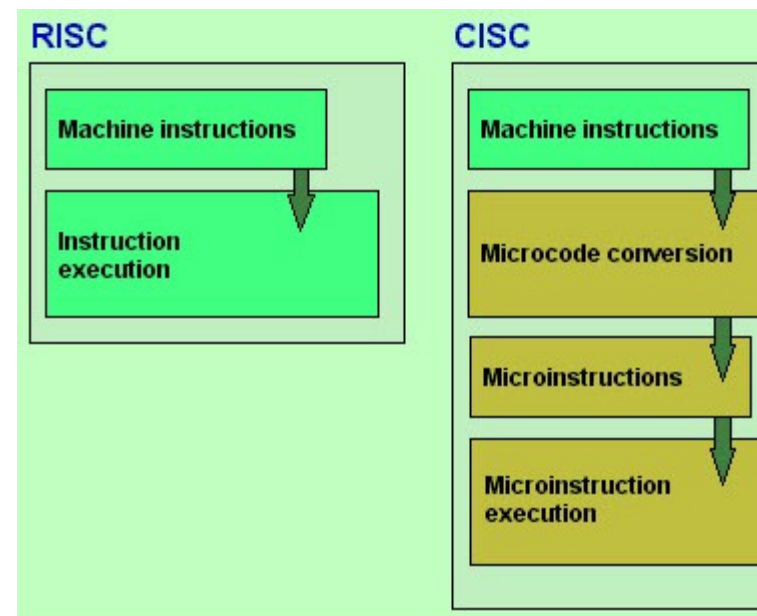


Fig 7(c). Instruction Execution.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

taxonomy of Computer Architecture

Classifies computer architectures based on the number of **instruction streams** and **data streams** processed simultaneously.

(Single Instruction, Single Data)

A single instruction operates on one data stream.
Sequential execution.

(Single Instruction, Multiple Data)

A single instruction operates on multiple data elements in parallel.
Used in vector processing, GPUs, etc.

(Multiple Instruction, Single Data)

Multiple instructions operate on the same data stream.
Rare in practice.

(Multiple Instruction, Multiple Data)

Multiple processors execute different instructions on different data streams.
Common in parallel systems.

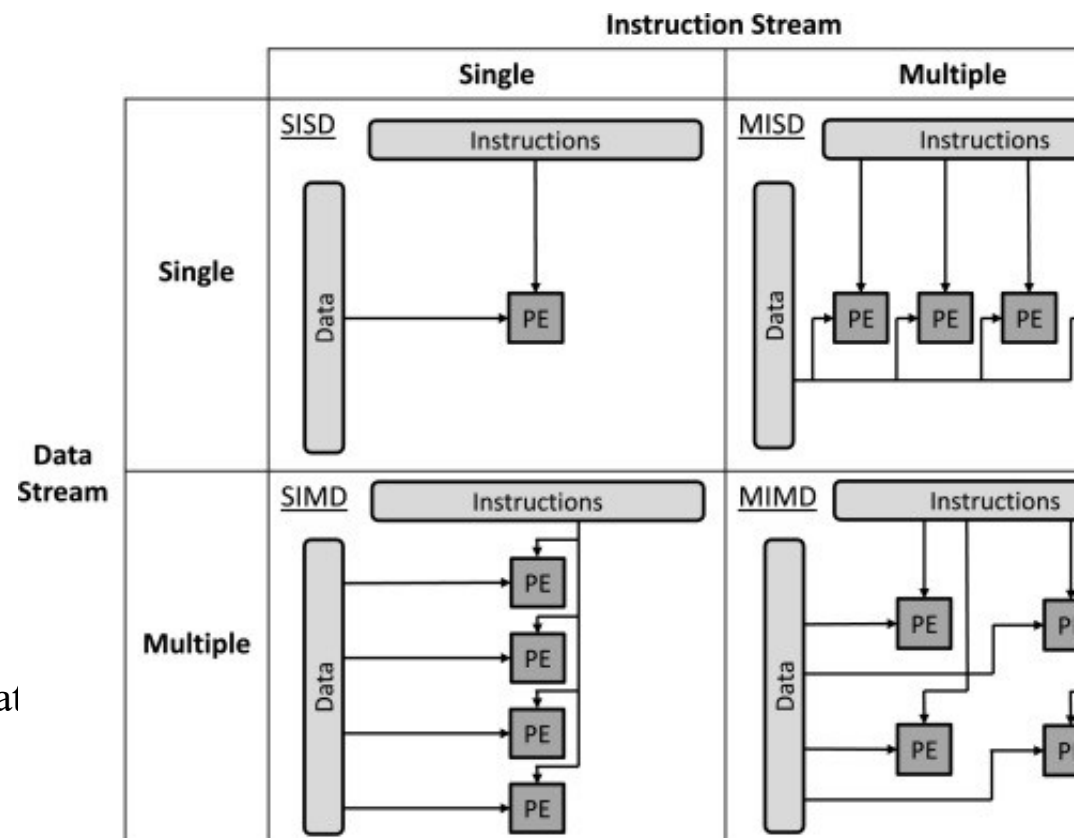


Fig 8(a). Flynn's Computer Architectures.

Verifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Hardware Software Co-Design

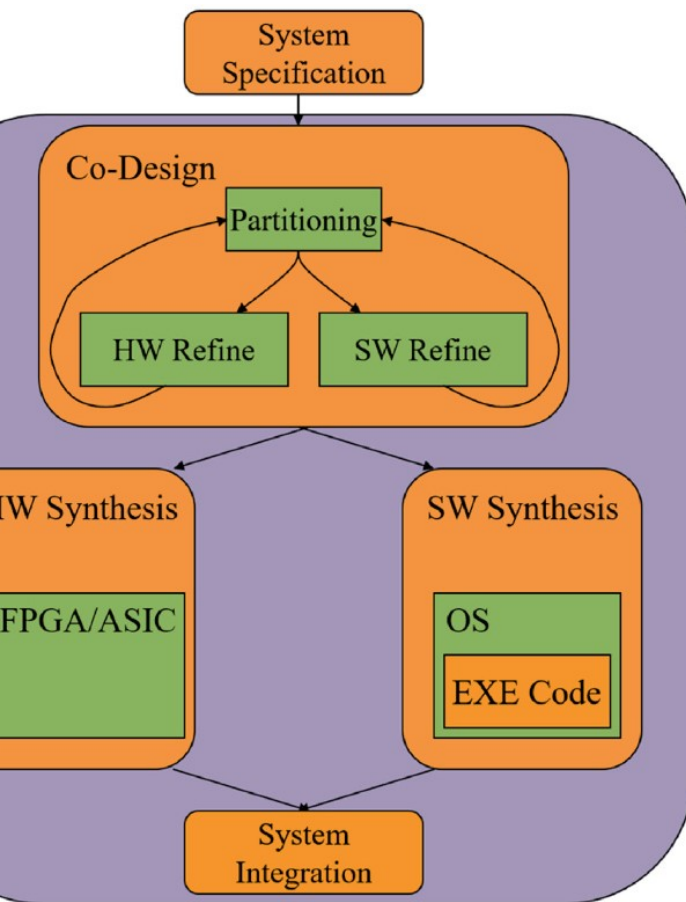


Fig 9(a). Hardware Software Co-Design.

PLDs

- **CPLD:** Fixed logic, predictable timing
- **FPGA:** Reconfigurable, parallel processing

Programming

- HDL: **Verilog / VHDL**
- Synthesized → Bitstream → Hardware
- Defines **hardware behavior**

HW-SW Interaction

- Memory-mapped I/O
- Bus (AXI/AHB)
- Interrupts, DMA

Role in Co-Design

- **Hardware:** Acceleration
- **Software:** Control logic

CPLD: Complex Programmable Logic Device.

FPGA: Field Programmable Gate Array.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Communication Protocols

2. **I2C** (Inter-Integrated Circuit)
Low speed on-board half duplex
Usage : Chip-to-chip, Sensors ...
Master / Slave communication

Universal Asynchronous Receiver/Transmitter

Asynchronous communication (no clock line)
Simple, widely used for serial communication

Serial Peripheral Interface

Synchronous (uses clock)
High speed, master-slave architecture
Used for sensors, flash memory

Inter-Integrated Circuit

Two-wire protocol (SDA, SCL)
Supports multiple devices (address-based)
Moderate speed, widely used in embedded systems

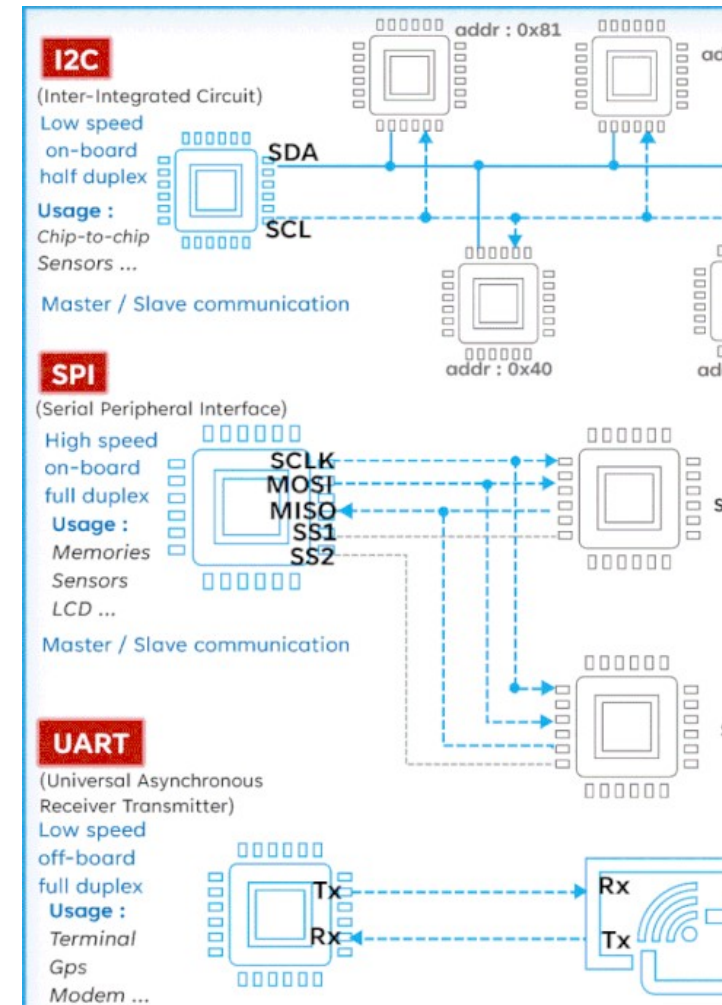


Fig 10(a). Communication Protocols.

Designing Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Constraints

Power Consumption

Crucial for **battery-operated systems**
Requires low-power design techniques (sleep modes, clock gating, etc.)

Performance (Timing Constraints)

Must **meet real-time deadlines**
Deterministic execution is essential

Memory Constraints

Limited **RAM and Flash**
Efficient memory management required

Need to **minimize hardware and production cost**
Optimizes component selection and design complexity

Form Factor

Compact design for portable/embedded applications
Space and component optimization required

Reliability & Robustness

- Must operate under **harsh conditions**
- Fault tolerance and error handling required

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Applications

Automotive Systems

Engine Control Unit (ECU)
Anti-lock Braking System (ABS)
Advanced Driver Assistance Systems (ADAS)

Medical Devices

ECG / patient monitoring systems
Medical equipment
Diagnostic systems

Consumer Electronics

Smart TVs, washing machines
Microwave ovens
Wearables (smartwatch, fitness band)

Communication Systems

Modems, routers
Mobile base stations
IoT devices

Aerospace & Defense

- Flight control systems
- Navigation systems
- Radar and surveillance

Industrial Automation

- PLCs (Programmable Logic Controllers)
- Robotics and automation systems
- Process control systems

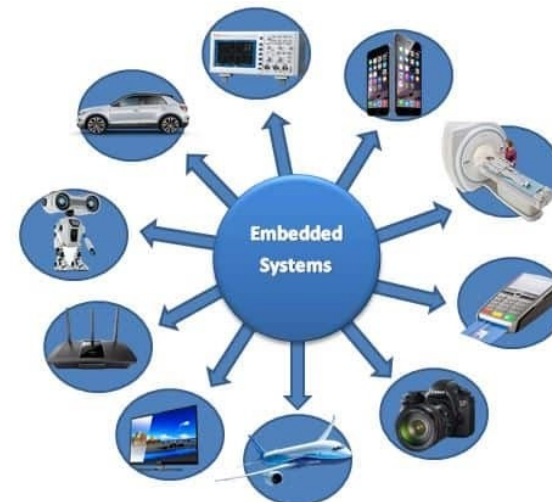


Fig 11(a). Applications of Embedded System.

Defining Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

Emerging Trends

Internet of Things (IoT)

Marked embedded devices with real-time data exchange
Smart homes, smart cities, industrial IoT

Computing & AI at Edge

Processing data **locally** instead of cloud
Enables **low latency and real-time intelligence**
Applications: vision systems, autonomous devices

Integration

Integrated AI accelerators (NPU, TPU)
Applications: speech recognition, image processing

Power & Energy-Efficient Design

Low power MCUs
Techniques: dynamic voltage scaling, sleep modes

Domain Acceleration

Configurable hardware for high-performance tasks
Applications: signal processing, AI inference

Autonomous & Smart Systems

- Drones, self-driving systems
- Robotics with real-time decision making

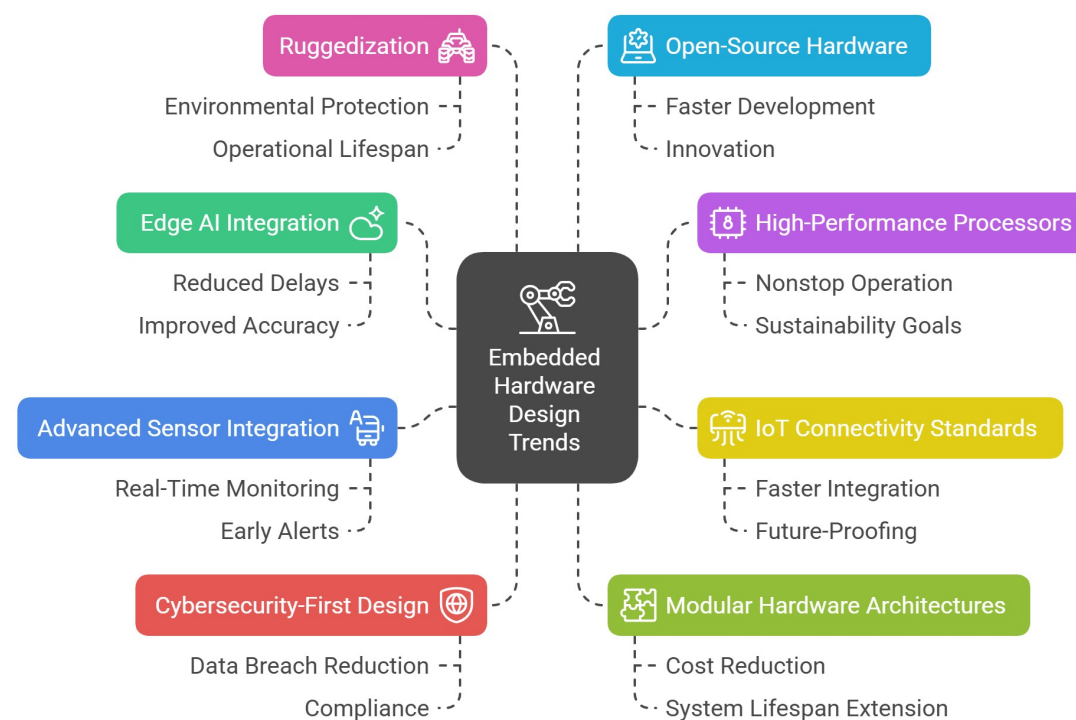


Fig 12(a). Emerging Trends in Embedded System.

Identifying Embedded Systems: Components, Applications, Design

- Dr. Arun Kumar

