

Distributed Intelligence Systems

Serial communication, Introduction to microcontroller topologies (star, ring, bus),

Serial Communication

Serial communication is a primary method for microcontrollers to exchange data with other devices such as sensors, computers, peripherals or other microcontrollers.

In Serial communication data is transmitted one bit at a time over a single wire or channel. It's commonly used in embedded systems due to its simplicity and efficiency.

Bits are sent sequentially in Serial Communication over a single channel or wire. This is in contrast to parallel transmission, where multiple bits are sent simultaneously over multiple channels. Serial transmission is widely used in communication systems due to its starkness and cost-effectiveness, especially for long-distance communication.

Serial Communication

The opportunity cost of serial Communication is parallel Communication in the transfer of data between CPU, memory and subsystems (I/O) within a microcontrollers.

Serial data connections are useful because only one or two signal wires are needed, compared with at least eight data lines for a parallel bus plus control signals.

Serial Communication— cheaper, slower and more complex hardware.

Parallel Communication — faster, expensive and simple



10011101

1 → 0 → 0 → 1 → 1 → 1 → 0 → 1

MicrocontrollerBoard.com



10011101

MicrocontrollerBoard.com

- It needs more wires and therefore can be more expensive to implement
- The greater number of wires limit it to shorter transmission distances
- It is susceptible to clock skew, which limits the speed of transmission to the slowest of the links
- Crosstalk is also an issue due to the proximity of the wires

Serial Communication

Serial Communication Basics

Serial communication transmits data **bit by bit over a single line or pair of lines**, unlike parallel communication which sends multiple bits simultaneously. It is widely used in microcontrollers because it reduces pin count and wiring complexity. Common serial communication protocols include **UART (Universal Asynchronous Receiver/Transmitter)**, **SPI (Serial Peripheral Interface)**, and **I²C (Inter-Integrated Circuit)**.

Key advantages of serial communication:

- Fewer physical connections, simplifying PCB and wiring layouts.
- Ability to communicate over longer distances compared to parallel communication.
- Compatibility with standard communication peripherals on microcontrollers.

Microcontroller Network Topologies

The physical or logical arrangement of devices in a network is called a **topology**. The selected topology affects **data flow, fault tolerance, scalability, efficiency and complexity** of the system.

1. Star Topology

In a **star topology**, all microcontrollers or devices are connected to a **central hub or master node**.

Data from any device passes through the central hub to reach its destination.

Characteristics:

- Simple to design and troubleshoot.
- Failure of one node does not affect others.
- Failure of the central hub stops communication for the entire network.
- Well-suited for systems with a clearly defined master-slave structure for serial communication like UART or SPI with a central controller.

2. Ring Topology

In a **ring topology**, each microcontroller is connected to two neighbors, forming a closed loop. Data travels in **one or both directions around the ring** until it reaches the target device.

Characteristics:

- Each device acts as a repeater, forwarding data along the ring.
- Can allow orderly access to the bus (e.g., token passing).
- Single node failure can disrupt the entire network unless redundancy mechanisms are implemented.
- Suitable for deterministic and time-critical communication in embedded systems.

3. Bus Topology

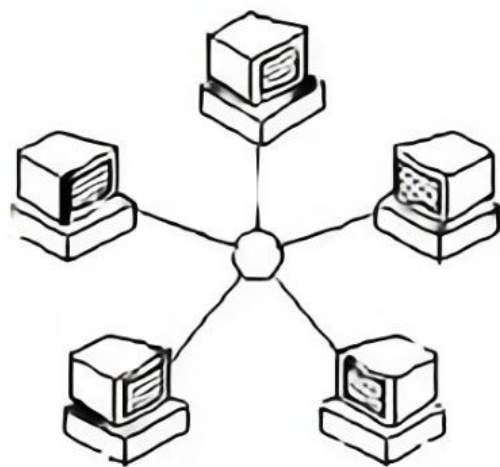
In a **bus topology**, all devices share a **common communication line** (serial bus). Only one device transmits at a time while others listen. **Characteristics:**

- Simple layout; requires fewer cables than star topology.
- Requires careful management of data collisions using protocols like I²C or CAN (Controller Area Network).
- Failure in the main bus line or improper termination can disrupt the entire network.
- Efficient for systems with multiple devices exchanging data over a shared serial line.

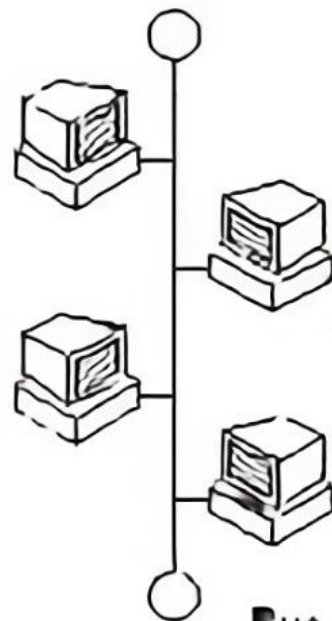
Summary

Serial communication combined with appropriate topologies allows microcontrollers to **efficiently interface and coordinate** tasks. Choosing between star, ring, and bus topologies depends on factors such as:

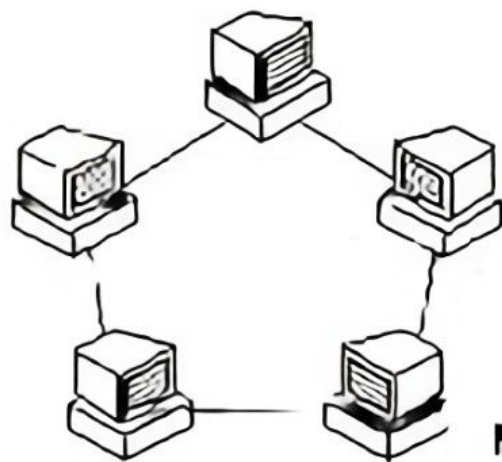
- Number of devices
- Criticality of continuous operation
- Ease of expansion and maintenance



Star



Bus



Ring

Types of Serial Communication in Microcontrollers

1. UART (Universal Asynchronous Receiver/Transmitter)

- **Asynchronous:** No clock signal is shared.
- Uses **TX (Transmit)** and **RX (Receive)** lines.
- Common baud rates: 9600, 115200, etc.
- Widely used for debugging and communication with PCs.

2. SPI (Serial Peripheral Interface)

- **Synchronous:** Uses a shared clock.
- Full-duplex, high-speed.
- Uses **MOSI, MISO, SCLK**, and **SS/CS** lines.
- Ideal for short-distance, high-speed communication (e.g., SD cards, displays).

3. I2C (Inter-Integrated Circuit)

- **Synchronous**, multi-master, multi-slave.
- Uses only **2 wires**: SDA (data) and SCL (clock).
- Slower than SPI but supports more devices with fewer pins.

4 USB (Universal Serial Bus)

- More complex, but allows microcontrollers to act as USB devices (e.g., keyboards, serial ports).
- Requires USB-capable microcontrollers (e.g., STM32, ATmega32U4).

5 CAN (Controller Area Network)

- Used in automotive and industrial applications.
- Robust, supports long distances and multiple nodes.

References

- 1 [Serial Communication with Microcontrollers | Microcontroller Tutorials](#)
- 2 [PIC microcontroller serial communication tutorial](#)

