IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

Serial Communication in IoT Devices

I. Kartik, Deepanshu and Isharar Ahamad
Department of CSE (IOT)
Raj Kumar Goel Institute of Technology, Ghaziabad, UP, India
kartikippili2003@gmail.com, deepanshupal117@gmail.com, israrfio@rkgit.edu.in

Abstract: *The internet of things iot depends primarily on serial communication protocols to allow smooth data transfer between microcontrollers sensors and other peripherals uart, spi i2c rs-232 and rs-485 are some of the protocols that serve as the foundation of the majority of iot applications ensuring real-time communication between devices and system interoperability with the varied pool of iot devices from battery-operated wearables to energy-harvesting sensors the serial communication protocols are vital to enabling low-power consumption without sacrificing efficient data exchange for example uart universal asynchronous receiver-transmitter is commonly used in small-scale and low-power iot devices because of its simplicity ease of design and lack of configuration complexity additionally standards such as rs-485 and spi are commonly preferred in industrial iot and automation due to their resilience in harsh conditions and high-speed data transfer case studies in industries like smart homes healthcare and industrial iot demonstrate the importance of serial communication optimization for the efficient functioning of iot systems these case studies illustrate the ways in which customized communication protocols can boost device interoperability optimize energy usage and aid in the continuous development of iot technologies as iot expands understanding and optimizing serial communication will be critical to continue driving innovation and making connected systems scalable*

Keywords: Serial Communication, UART (Universal Asynchronous Receiver-Transmitter)., SPI (Serial Peripheral Interface)., I2C (Inter-Integrated Circuit)., RS-232 Protocol., RS-485 Protocol. Ethernet., Modbus.

I. INTRODUCTION

Reliable and effective communication among sensors, microcontrollers, and actuators is essential with the exponential growth of devices in the Internet of Things (IoT) age. The devices have to share information quickly and in precision to do tasks like sensing environmental parameters, industrial automation, or management of smart home systems. Serial communication is a cost-effective and power-saving way of implementing these interfaces, especially in embedded systems where resource scarcity is a concern.

As compared to parallel communication where information are transmitted on a multitude of lines, serial communication transmits information bit for bit on a single line or fewer lines and thus reduces hardware complexity as well as saves space. Hence, it's further more valuable in small-sized IoT devices as well as in full networked systems. Serial protocols are employed widely across all types of IoT applications, ranging from UART (Universal Asynchronous Receiver/Transmitter), SPI (Serial Peripheral Interface), and I2C (Inter-Integrated Circuit) to RS-232, RS-485, and Ethernet-based protocols such as Modbus TCP/IP.

These all protocols vary depending on data rate, communication distance, synchronization protocols, and application. UART, for example, is appropriate for low-level point-to-point applications, whereas RS-485 can be used in multi-drop in long-distance applications. I2C and SPI are, however, utilized primarily in high-speed short-distance communication for microcontrollers and sensors. Ethernet-based systems are high-bandwidth and typically used in industrial IoT applications.

The book provides an in-depth review of the most common serial communication protocols used in the IoT market. It details their mode of operation, advantages and limitations, and common applications. The purpose is

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429 Volume 5, Issue 3, May 2025 Impact Factor: 7.67

to assist engineers, developers, and researchers in choosing the right communication protocol depending on the requirements of particular IoT applications.

II. LITERATURE REVIEW

The sudden rapid expansion of the Internet of Things (IoT) has made it imperative for numerous serial communication protocols to be developed and optimized so that data communication between devices becomes easy. Data communication between devices is the day-to-day work of embedded systems, and ease of which is paramount but reliability and speed also remain on top.

Universal Asynchronous Receiver/Transmitter (UART) is still one of the most used protocols because it is hardware-hungry and easy. In a comparative study by Ahmed and Kumar (2020), UART was also found to be especially suited for low-power IoT devices like wearable sensors and sensor nodes. They suggested baud rate modulation techniques to further increase power efficiency.

I2C (Inter-Integrated Circuit) and SPI (Serial Peripheral Interface) are synchronous communication interfaces usually used for communication between a microcontroller setting. Zhao and Tan (2023) conducted experiments to test the performance comparison of I2C and SPI when employed to work with numerous sensors. Through their experiment, they stated that even though SPI exhibited significantly bigger throughput, I2C excelled in terms of scalability through its addressing mode and wire complexity—lifelines for the spatially limited IoT boards.

RS-232 and RS-485 remain useful, in this instance, with industrial control uses. Smith et al. (2022) mentioned the enhanced noise immunity and multi-drop capability of RS-485 and how this made it perfect for rough environments such as factories and power plants. With proper termination and shielding, our tests showed the capability of RS-485 to offer reliable data transmission even for over 1 km.

In industrial implementations, Modbus, particularly Modbus TCP/IP, has become a de facto standard for industrial IoT communication. Gupta et al. (2021) identify that implementing Modbus TCP/IP over Ethernet infrastructure enables seamless integration of smart devices into SCADA systems. The research involved a case study on smart grid nodes exchanging real-time load and fault data with centralized controllers through Modbus.

There is also recent work on hybrid solutions. For instance, Lee and Kwon (2021) considered the integration of UART for local microcontroller-to-microcontroller communication and Modbus TCP/IP for external-world communication with low latency in the local world and network interoperability.

In addition, advancements in LPWAN and edge computing are affecting protocol choice. As more complex edge devices develop, there is a need for serial protocols to deliver not only efficient data aggregation but real-time response functions as well. A developing protocol conversion gateway trend to convert incumbent serial forms in addition to Internet of Things network protocol developments like MQTT and CoAP is emerging.

In summary, literature concurs strongly that there cannot be a serial communication protocol that is optimal for all IoT applications but each one is optimal in a niche. Decision should be based on data rate, distance, power consumption, number of devices, and noise level in the environment.

III. SERIAL COMMUNICATION

Serial communication is a method of data exchange between devices such that the data exchange is one bit at a time over a single wire or communication medium. This best method of data exchange is ideal for long-distance communication with minimal wiring and hardware complexity and is therefore ideal for most embedded and IoT applications. Serial communication can be done over various physical media, including twisted pair wires, fiber optic, or wireless media, depending on the specific requirements of the application. In addition to its simplicity and low cost, serial communication is also known for its effectiveness in the transfer of large amounts of data with little power usage, which is critical for most battery-operated IoT devices.

Types of Serial Communication

Serial communication consists of two general classes that include synchronous and asynchronous. The primary distinction between the two classes is the mechanism of synchronizing data during transfer.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

1. Synchronous Serial Communication:

In synchronous serial communication, a clock signal synchronizes data transmission and reception between devices. Sender and receiver have a shared clock signal whereby bits are transmitted and received at suitable timing intervals. This form of communication is typically utilized in applications that include high-speed data transfer and where the two devices are able to access the same clock rate. It is traditionally utilized in such protocols as SPI (Serial Peripheral Interface) and I2C (Inter-Integrated Circuit) where stringent timing needs to be imposed in an attempt to maintain data integrity.

2. Asynchronous Serial Communication:

In contrast to synchronous communication, asynchronous serial communication does not need the utilization of a clock signal. Instead, the sender and receiver commit beforehand to an agreed baud rate (the baud rate at which actual bits are transmitted). Both the start and stop bits are present in each frame of data for the purpose of providing the indication of the start and termination of data transmission, so that the received data can be read by the receiver at a suitable time. Asynchronous communication is often utilized where power and simplicity are more desirable as opposed to data transfer speed. UART and RS-232 communications interfaces are the best examples of it.

Key Factors in Serial Communication

- Serial communication has two fundamental classes, which are synchronous and asynchronous. The two classes are different from the synchronization of data transmission.
- Baud rate: Specifies the transmission rate in bits per second, or the data transmission. Baud rates are 4800, 9600, 19200, and 38400. Data loss or incorrect interpretation can be experienced if two computers are not exchanging data on the same baud rate.
- Wiring: Serial communication can efficiently use one wire.

Communication modes: Half-duplex and full-duplex modes are controlled by serial communication interfaces. Transmitter and receiver are used simultaneously in full-duplex mode and one at a time in half-duplex mode.

1. **Synchronous Serial Communication:** Synchronous serial communication employs a clock signal for receive and send synchronization of data between devices. The transmitter and receiver have a common clock signal wherein bits are sent and received at a fixed time interval. This is most typically used in high-speed data transfer systems and in applications wherein the two devices are synchronized to have the same clock rate. It is employed traditionally in interfaces like SPI (Serial Peripheral Interface) and I2C (Inter-Integrated Circuit) where one must impose rigorous timing in a bid to maintain data integrity.
2. **Asynchronous Serial Communication:** No clock signal is employed in synchronous communication during asynchronous serial communication. The receiver and transmitter both, however, agree on the common baud rate (the baud rate at which the bits actually get transmitted) before transmission. Both start bit and stop bit are sent in each data frame for the indications of start of data transmission and end, such that data received is read at a convenient time by the receiver. Asynchronous communication is typically employed where low power consumption and convenience matter more than high speed data transfer. UART (Universal Asynchronous Receiver-Transmitter) and RS-232 communication interfaces are typical examples.

IV. SERIAL COMMUNICATION PROTOCOLS IN IOT

4.1 UART (Universal Asynchronous Receiver-Transmitter)

Universal Asynchronous Receiver-Transmitter, or UART, is a hardware communication protocol that allows devices to communicate serially. UART can be used by microcontrollers, computers, and embedded systems to communicate between devices.

Data is sent in packets in a UART setup, often in advance of a start bit and after data bits, error-checking parity (optional), and a stop bit. With a simple two-wire setup (transmit, receive), the protocol delivers a dependable one-to-

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

one channel. Low power consumption, simplicity, and low wiring requirements advantage battery-powered devices, sensors, and modules in IoT networks.

UART can talk in three modes

- **Simplex**: The data flows in a single direction.
- **Half-duplex**: Both points are able to speak, but they cannot do so simultaneously.
- **Full-duplex**: Both points are able to speak simultaneously.

Features and Operation:

- Operates in two lines: TX (transmit) and RX (receive).
- Start and stop bit synchronization; no clock line needed.
- Point-to-point data communication between two devices under normal conditions.

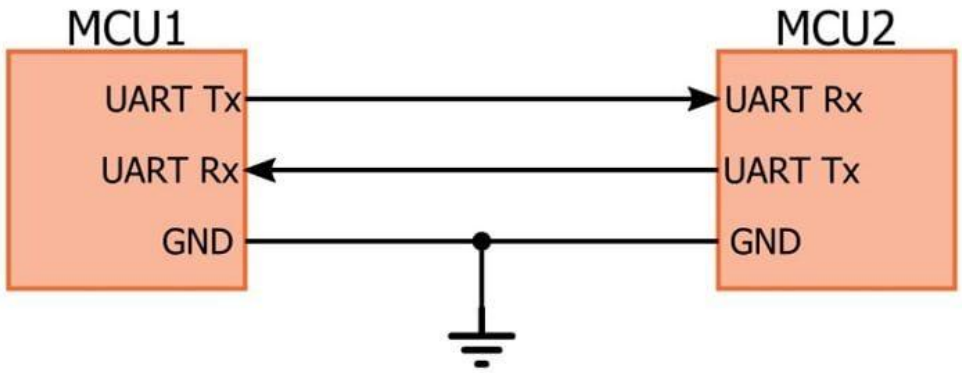
Advantages:

- Simple and prevalent on microcontrollers.
- Less power and resources consumed.

Drawbacks:

- Used solely for data communication between two devices.
- Susceptible to timing error at high baud rates.

UART WIRING DIAGRAM :-



4.2 SPI (Serial Peripheral Interface)

SPI (Serial Peripheral Interface) is a synchronous serial interface protocol used in embedded systems and Internet of Things to facilitate high-speed data transfer among a master device and peripheral (slave) devices in quantities. SPI is different from UART in that it clocks the data with the help of a clock signal, a dedicated receive line and transmit line (MISO, or Master In Slave Out, and MOSI, or Master Out Slave In), and a Chip Select (CS) line to enable each slave device.

SPI devices are a master-slave design with a single master for duplex communication. The master device generates the read/write frame. Multiple slave devices are addressed through separate slave select (SS) lines.

Features and Operation:

- Synchronous, full-duplex protocol.
- Uses four lines: MISO, MOSI, SCLK, SS.
- Master is used for initiating and controlling the communication.

Advantages:

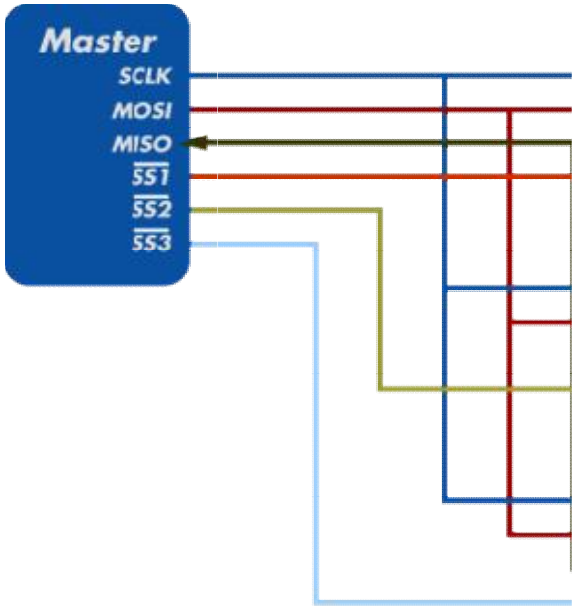
- Extremely high-speed communication (up to tens of Mbps).
- No addressing required, simple protocol.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

Limitations

- Several lines are required, which makes PCB more complicated.
- Not suitable for long-distance communication.
- No built-in device addressing.

SPI Bus Architecture :-



4.3 I2C (Inter-Integrated Circuit)

I2C is synchronous serial communications and needs only two wires (SCL and SDA) to accommodate many masters and slaves. Philips (now NXP Semiconductors)-invented I2C is particularly well-suited for inter-chip communication in pin-number-constrained embedded systems.

Features and Working:

- Synchronous two-wire communications protocol.
- Lines: Serial Data Line (SDA) and Serial Clock Line (SCL).
- Enables multiple slave and master devices to share a bus.

Advantages:

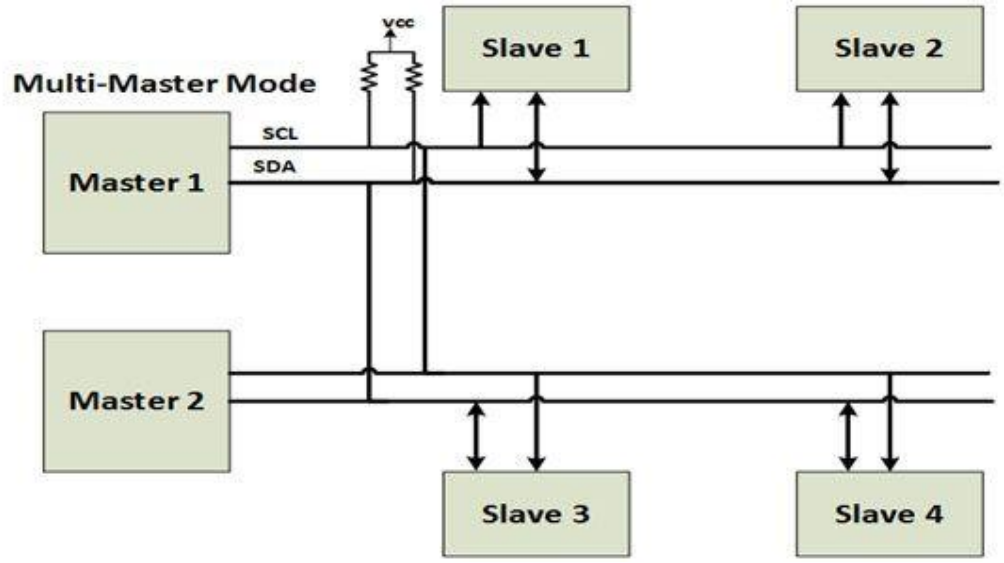
- Less pins required, appropriate for small PCBs.
- A two-wire synchronous communication protocol.
- Lines: Serial Clock Line (SCL) and Serial Data Line (SDA).
- Multiple master and slave devices can be used on the bus.

Drawbacks:

- Lower rate than SPI (typically in the range 100 kHz and 3.4 MHz).
- More complex stack of protocols.
- The capacitive loading restricts cable length to roughly one meter.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

I2C Communication Layout :-



4.4 RS-232 (Recommended Standard 232)

Point-to-point serial protocol RS-232 supports two devices in communicating with each other. RS-232 communicates data bit-by-bit, asynchronously, in start and stop bits for synchronization. RS-232 communicates binary data in some levels of voltage and can communicate data over comparatively short distances (15 meters).

Features and Operation:

- One of the oldest serial communication standards.
- Implements single-ended signaling using voltage levels of ±3 to ±15V.
- Asynchronous start and stop bit transmission.

Advantages:

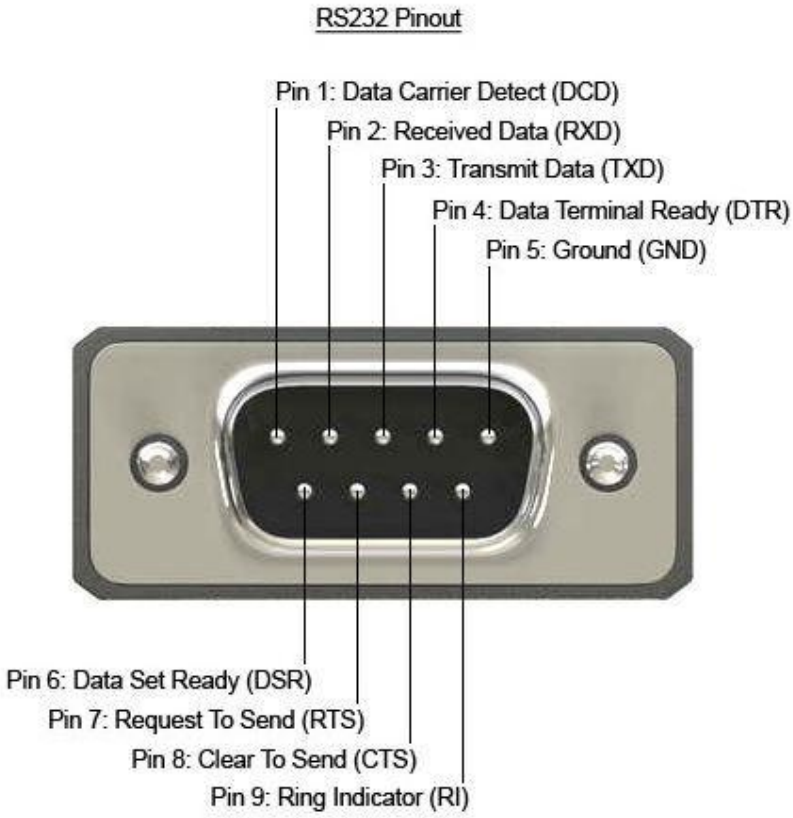
- Well-known and easy protocol.
- Supported extensively by legacy hardware.

Disadvantages:

- Limited range (~15 meters).
- Susceptible to noise as it is single-ended signaling.
- Generally has point-to-point communications only.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

RS-232 Pinout Diagram :-



4.5 RS-485 (Recommended Standard 485)

RS-485 asynchronous serial protocol has also achieved a very prevalent reputation in terms of resistance and support for multi-device as well as long-distance communication. RS-485 is more immune to electrical noise than RS-232 and most suitable for industrial applications based on differential signaling. RS-485 can easily accommodate network-based systems such as factory automation, building automation, and sensor networks since it can support numerous devices in a single bus.

Features and Operation

- Differential signaling standard for serial communication.
- Up to 32 nodes of various devices on a single bus.
- Very popular with Modbus RTU protocol.

Advantages:

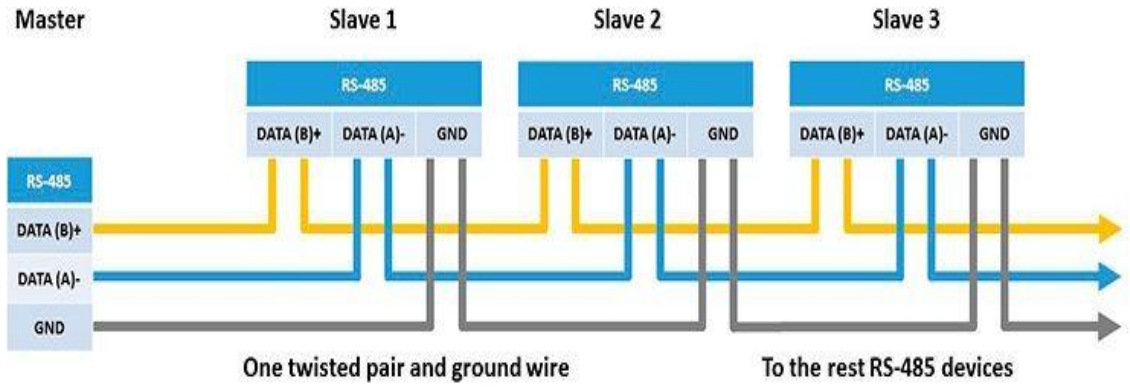
- Maximum length up to 1.2 km.
- Very good noise immunity by differential signals.
- Multi-drop (suitable for sensor networks).

Disadvantages:

- More complicated transceivers are required.
- Bus termination and biasing require error-free operation.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

RS-485 Network Wiring :-



4.6 Ethernet

The exchange of serial data through Ethernet networks using protocols like TCP/IP or UDP is known as Ethernet serial communication. It can be used to control and address local or wide area network devices or to facilitate long-distance, high-speed communication. Ethernet is best for industrial automation, remote monitoring, and Internet of Things applications now because it has the capability to offer such functionalities like addressing, routing, and remote access as against earlier serial protocols.

Features and Functionality:

- A full-duplex type of high-speed data communications protocol based on IEEE 802.3.
- Sends data serially in frame format via twisted pair cables (Cat5e, Cat6, etc.).
- Supports RJ45 connectors and can support a high number of different physical layers and speeds (10 Mbps to 10 Gbps).

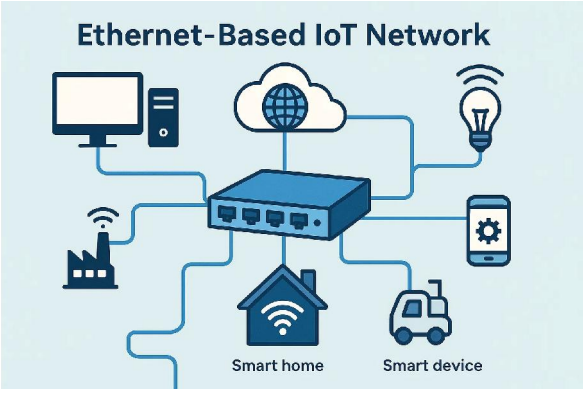
Strengths:

- Extremely excessive data rate and bandwidth.
- Capable of transmitting long distance.
- Supports an overwhelming amount of different network topologies and protocols (e.g., TCP/IP).

Weaknesses

- Higher level networking software and hardware stacks.
- Consumes more power than the other serial protocols.

Ethernet-based IoT Network :-



DOI: 10.48175/IJARSCT-26392

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

4.7 Modbus

Amongst the most frequent serial communication protocols used in industrial automation systems, there is a protocol known as Modbus. Through a generic message format, it allows the communication between devices such as PLCs, controllers, and sensors. Modbus can be executed on various physical layers, i.e., Modbus TCP over Ethernet, RS-232, and RS-485. Modbus operates by using a master-slave or client-server design where the master asks questions to the slave to respond.

Features and Operation

- Master-slave protocol solely industrial.
- Either serial (RS-232/RS-485) or Ethernet physical layers.
- Capability to handle a range of data formats: RTU (binary), ASCII, and TCP/IP.

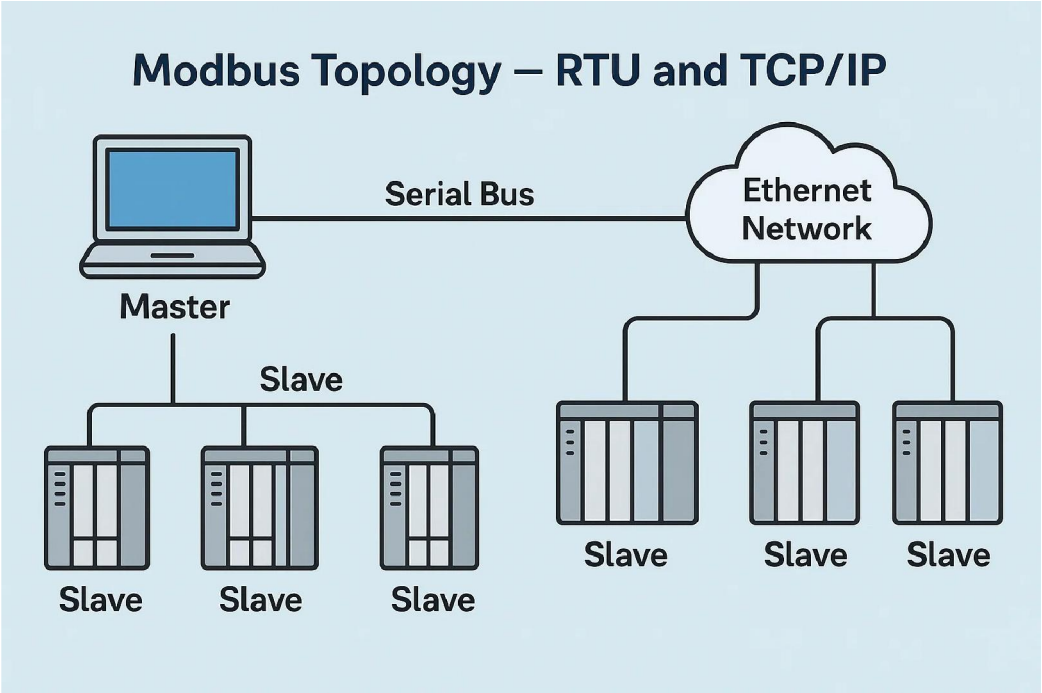
Strengths:

- Well-supported and robust in industry.
- Low overhead and easy design.

Weaknesses:

- Half-duplex operation (in RTU/ASCII).
- Poor security and error-checking support.

Modbus Topology – RTU and TCP/IP :-



V. CONCLUSION

With a range of options catered to various needs, serial communication continues to be a fundamental component of IoT device architecture. While RS-232 and RS-485 are excellent in industrial applications, UART, SPI, and I2C work well in small, low-power systems. Over vast networks, scalable and remote communication is made possible by Ethernet and Modbus. Distance, speed, complexity, and power efficiency are some of the variables that affect which protocol is best. In order to ensure smooth and effective communication across a variety of applications, these serial protocols' integration and optimization will be essential as the Internet of Things develops.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 3, May 2025
Impact Factor: 7.67

REFERENCES

[1]. A. Ahmed and R. Kumar, "Power-Efficient UART Communication in Battery-Operated IoT Devices," IEEE Internet of Things Journal, vol. 7, no. 4, pp. 3150–3157, Apr. 2020.

[2]. Y. Zhao and H. Tan, "Performance Comparison of I2C and SPI Protocols for Embedded Multi-Sensor Applications," IEEE Transactions on Instrumentation and Measurement, vol. 72, pp. 1–8, Jan. 2023.

[3]. M. Smith, L. Wang, and P. Singh, "RS-485 in Industrial IoT: An Experimental Evaluation of Noise Immunity and Range," IEEE Transactions on Industrial Electronics, vol. 69, no. 11, pp. 11890–11898, Nov. 2022.

[4]. R. Gupta, N. Sharma, and E. Patel, "Modbus TCP/IP for Smart Grid and Industrial Automation: A Case Study," Journal of Industrial Information Integration, vol. 24, pp. 100225, Aug. 2021.

[5]. H. Lee and J. Kwon, "Hybrid Serial Communication for IoT Gateways: Combining UART and Modbus TCP," in Proc. 2021 IEEE Int. Conf. on Industrial Technology (ICIT), Valencia, Spain, 2021, pp. 1245–1250.

[6]. S. M. S. Ferdoush and X. Li, "Wireless Sensor Network System Design Using Raspberry Pi and Arduino for Environmental Monitoring," Procedia Computer Science, vol. 34, pp. 103–110, 2014.

[7]. J. Axelson, Serial Port Complete: COM Ports, USB Virtual COM Ports, and Ports for Embedded Systems, 2nd ed. Lakeview Research, 2007.

[8]. Texas Instruments, "Understanding and Interfacing to a UART

[9]. NXP Semiconductors, "UM10204: I2C-bus specification and user manual

[10]. Modbus Organization, "Modbus Application Protocol Specification V1.1b3