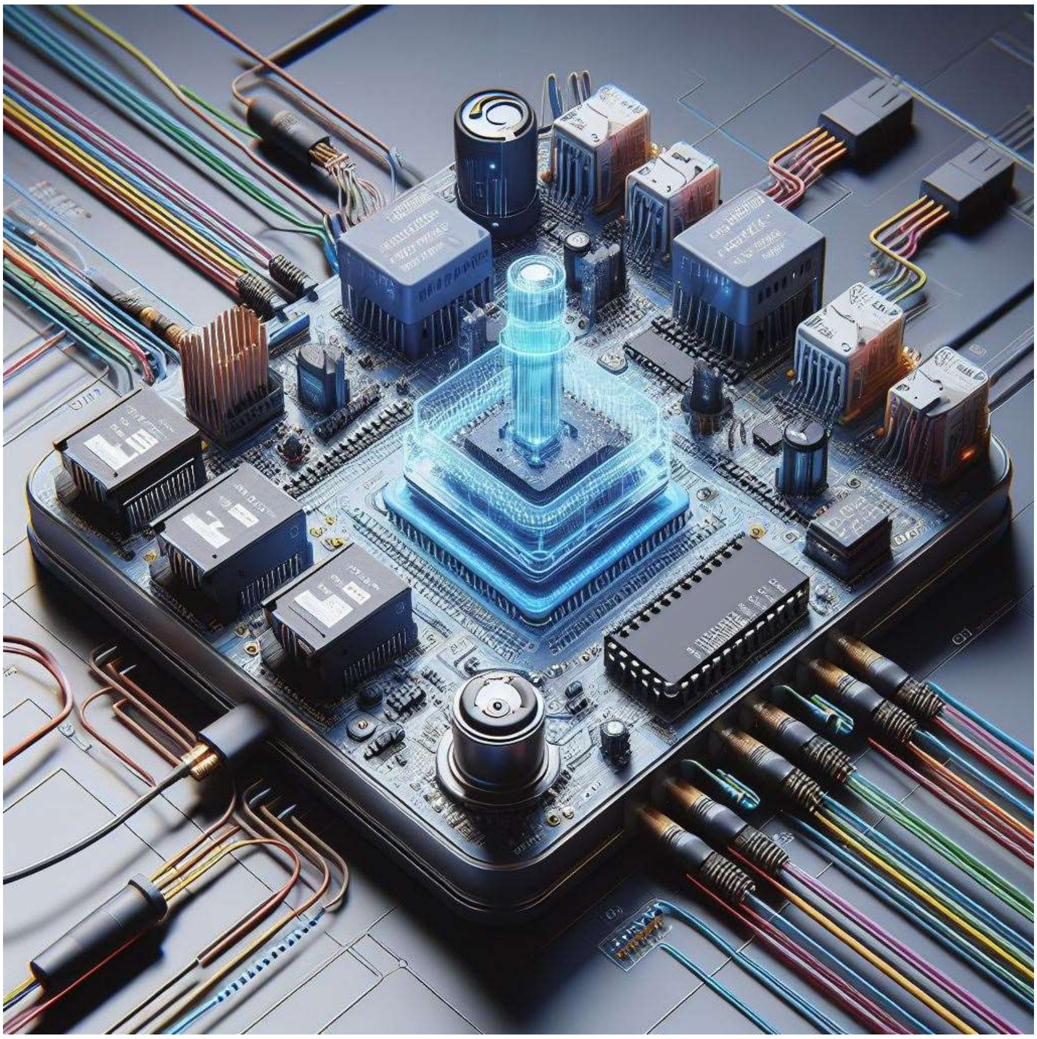
Alexandria University
Faculty of Engineering
Electrical Engineering Department



Communication Protocols.

⇨ UART, SPI, I2C, and CAN

Name / Khaled Ahmed Zaki Saad
ID / 20010513
Section / 6

Alexandria University

Faculty of Engineering

Electrical Engineering Department

Table of Contents

Introduction

Communication protocols are essential for seamless data exchange in embedded systems. In this report, we'll explore four key communication protocols: UART, SPI, I2C, and CAN.

Let's define some terms:

1. Half-Duplex:

- Definition: Half-duplex is a type of communication in which data can flow back and forth between two devices, but not simultaneously
- Explanation: Each device in a half-duplex system can send and receive data, but only one device can transmit at a time. An example of a half-duplex device is a walkie-talkie or a CB (citizens band) radio.
- Use Case: Walkie-talkies allow users to communicate back and forth on a specific radio frequency, but only one person can speak at a time.

2. Full-Duplex:

- Definition: Full duplex is a mode of communication in which data is **simultaneously transmitted and received** between stations.
- Explanation: Unlike half-duplex, where devices take turns, full-duplex allows both devices to **communicate simultaneously**. It typically uses **two separate channels** (wires or wireless frequencies) for transmission and reception.
- Use Case: Full-duplex communication is common in **phone calls**, **video meetings**, and **Ethernet networks**.

3. Synchronous Communication:

- Definition: Synchronous communication occurs between two or more people in **real-time**. It involves immediate exchange of information.
- Explanation: Participants are present simultaneously, and messages are exchanged instantly. Examples include **face-to-face conversations**, **phone calls**, and **live presentations**.

- Use Case: Real-time brainstorming sessions or urgent discussions benefit from synchronous communication.

4. Asynchronous Communication:

- Definition: Asynchronous communication is any type of communication where there is a **time lag** between providing information and receiving responses.
- Explanation: It doesn't happen in real-time. Participants can respond at their own pace. Examples include **email**, **Slack messages**, and **webinars**.
- Use Case: When teams work across different time zones or have varying schedules, asynchronous communication ensures flexibility.

5. Master/Slave:

- Definition: In master-slave communication, one device (the **master**) controls one or more other devices (the **slaves**). The master serves as the communication hub.
- Explanation: The master initiates and manages communication, while slaves respond to commands. Examples include **parallel ATA hard drive arrangements**, **database replication**, and **photography equipment**.
- Use Case: Master-slave relationships are common in various systems, but some organizations prefer more modern terms (e.g., **controller/peripheral**).

6. Arbitration:

- Definition: Arbitration is the process by which devices on a shared communication bus determine which device has the **right to transmit data**.
- Explanation: It prevents data collisions and ensures orderly communication. Different protocols use various arbitration methods (e.g., **CAN** uses bitwise arbitration).
- Use Case: Ensuring efficient and conflict-free communication in multi-device environments.

1. Universal Asynchronous Reception and Transmission (UART)

Overview

- **UART** (Universal Asynchronous Receiver/Transmitter) is a fundamental communication protocol used for serial data transmission.
- It is widely employed in embedded systems, microcontrollers, and various electronic devices.

Key Features

1. **Asynchronous Communication**:
    - UART operates without a clock signal, making it suitable for scenarios where precise timing is not critical.
    - Data is transmitted as a series of bits, with start and stop bits framing each byte.
2. **Full-Duplex Communication**:
    - UART allows simultaneous transmission and reception of data.
    - Devices can send and receive data independently.
3. **Baud Rate Control**:
    - The baud rate determines the data transfer speed.
    - Common baud rates include 9600, 19200, 115200, etc.

Applications

- **Serial Communication**: UART is commonly used for:
    - **Debugging**: Connecting a microcontroller to a PC for debugging purposes.
    - **Sensor Interfaces**: Interfacing with sensors, GPS modules, and RFID readers.
    - **Wireless Modules**: Communicating with Bluetooth or Wi-Fi modules.

Limitations

- **Point-to-Point**: UART supports communication between only two devices.
- **Lack of Addressing**: Unlike I2C or CAN, UART lacks built-in addressing mechanisms for multiple devices on the same bus.

2. Serial Peripheral Interface (SPI)

Overview

- **SPI** (Serial Peripheral Interface) is a synchronous communication protocol.
- It facilitates high-speed data exchange between a master device (usually a microcontroller) and multiple slave devices.

Key Characteristics

1. **Synchronous Communication**:
   - SPI relies on a clock signal (SCK) to synchronize data transmission.
   - Data is transmitted in full-duplex mode (simultaneous send and receive).
2. **Multiple Data Lines**:
   - **MISO (Master In Slave Out)**: Transmits data from slave to master.
   - **MOSI (Master Out Slave In)**: Transmits data from master to slave.
   - **SS/CS (Slave Select/Chip Select)**: Enables communication with specific slave devices.
3. **Short-Distance Communication**:
   - SPI is commonly used within a single PCB or short cable lengths.

Applications

- **Flash Memory**: SPI is used to interface with flash memory chips.
- **Display Modules**: Connecting TFT displays, OLED screens, or LED matrices.
- **Sensor Networks**: Communication with accelerometers, gyroscopes, and temperature sensors.

3. Inter-Integrated Circuit (I2C)

Overview

- **I2C** (Inter-Integrated Circuit), also known as **Two-Wire Interface (TWI)**, is a synchronous serial protocol.
- It enables communication between multiple devices using only two wires (SDA and SCL).

Key Characteristics

1. **Bidirectional Data Lines**:
    - **SDA (Serial Data Line)**: Carries data bidirectionally.
    - **SCL (Serial Clock Line)**: Provides the clock signal.
2. **Multi-Device Support**:
    - I2C allows multiple devices to share the same bus.
    - Each device has a unique address.
3. **Low-Speed Communication**:
    - Ideal for connecting sensors, EEPROMs, real-time clocks, and other low-speed peripherals.

Applications

- **Sensor Networks**: I2C is commonly used for:
    - **Temperature Sensors**: DS18B20, LM75, etc.
    - **Real-Time Clocks (RTCs)**: DS1307, DS3231, etc.
    - **EEPROMs**: Storing configuration data.

4. Controller Area Network (CAN)

Overview

- **CAN** (Controller Area Network) is a robust communication protocol designed for reliability and fault tolerance.
- Originally developed for automotive applications, it is now widely used in industrial automation and other domains.

Key Characteristics

1. **Differential Signaling**:
    - CAN uses differential signaling to reduce susceptibility to noise.
    - Twisted-pair cables carry complementary signals (CAN-High and CAN-Low).
2. **Message-Based Communication**:
    - Devices on the CAN bus exchange messages (frames) containing data and identifiers.
    - Broadcast communication (all devices receive the message).
3. **Error Detection and Correction**:
    - CAN includes mechanisms for error detection and automatic retransmission.
    - It ensures reliable data transmission even in noisy environments.

Applications

- **Automotive Systems**:
    - Engine control, anti-lock braking systems (ABS), airbags, etc.
- **Industrial Automation**:
    - Factory automation, robotics, process control.
- **Avionics**:
    - Aircraft communication networks.

Conclusion

Protocol	Full/Half Duplex	Synchronous/Asynchronous	Arbitration
UART	Full Duplex	Asynchronous	No built-in arbitration mechanism
I2C	Half Duplex	Synchronous	Arbitration based on bus access priority
SPI	Full Duplex	Synchronous	No built-in arbitration mechanism
CAN	Full Duplex	Synchronous	Arbitration based on message priority