INDIAN INSTITUTE OF TECHNOLOGY PATNA

EC3101: Microcontroller & Embedded Systems Lab

EXPERIMENT NO: 05

Implementation and Verification of Serial Communication Protocols

Name:	Shaurya Aggarwal
Roll No:	2301EE56
Group No:	2 (Thursday)

Aim:

Implementation and Verification of Serial Communication Protocols like UART, SPI, I^2C, USB using Arduino.

Software used:

Arduino IDE

Theory:

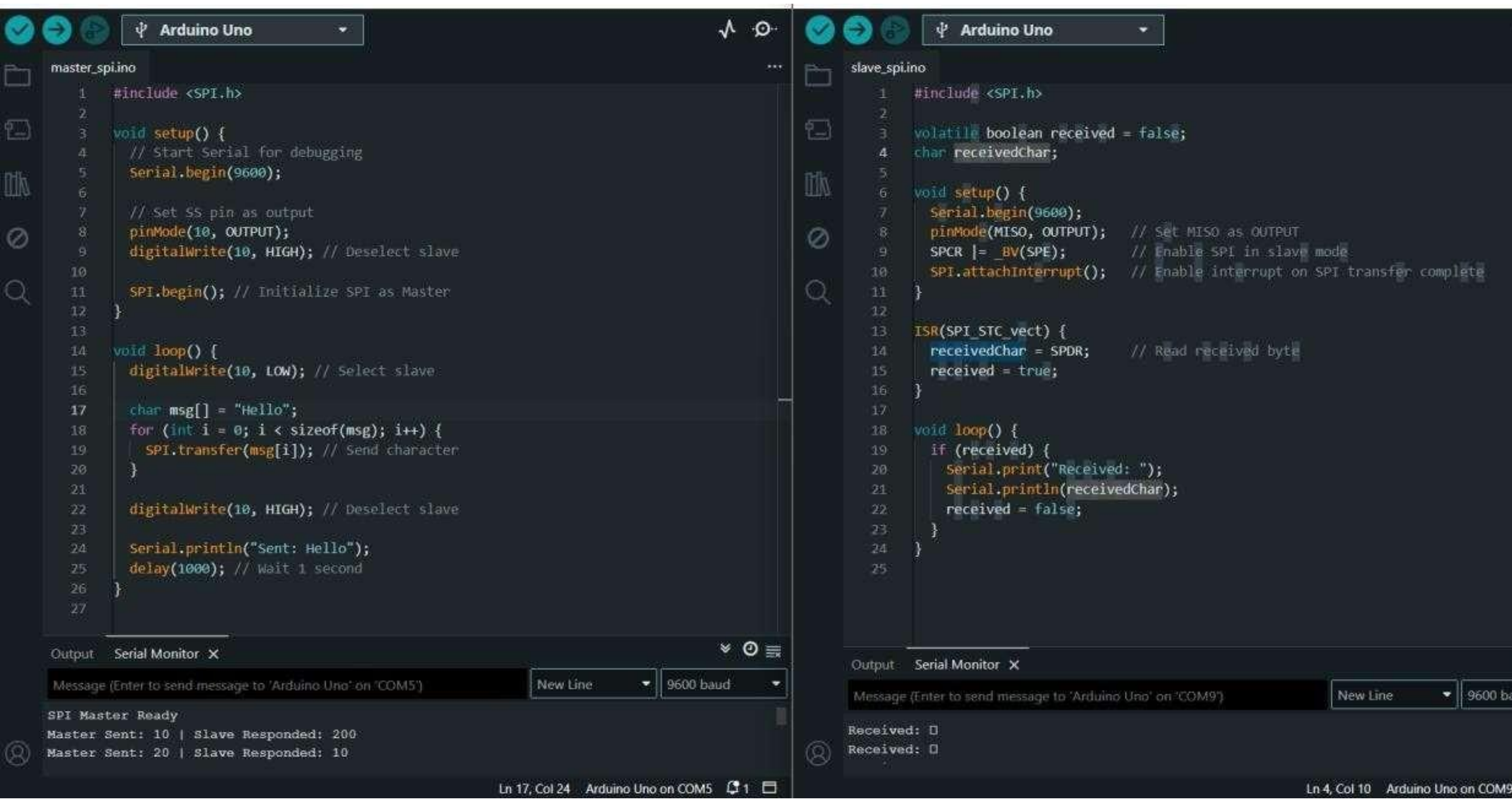
1. Serial Peripheral Interface (SPI)

The Serial Peripheral Interface (SPI) is a synchronous serial protocol designed for high-speed, short-distance communication, typically between a microcontroller and peripheral components like sensors or memory chips. It operates using a master-slave architecture, where a single master device coordinates the communication with one or more slave devices.

SPI communication relies on four main signal lines:

- MOSI (Master Out Slave In): Carries data from the master to the slave.
- MISO (Master In Slave Out): Carries data from the slave back to the master.
- SCK (Serial Clock): The clock signal generated by the master that synchronizes the data transfer.
- SS (Slave Select): A dedicated line for the master to select which slave device it wants to communicate with.

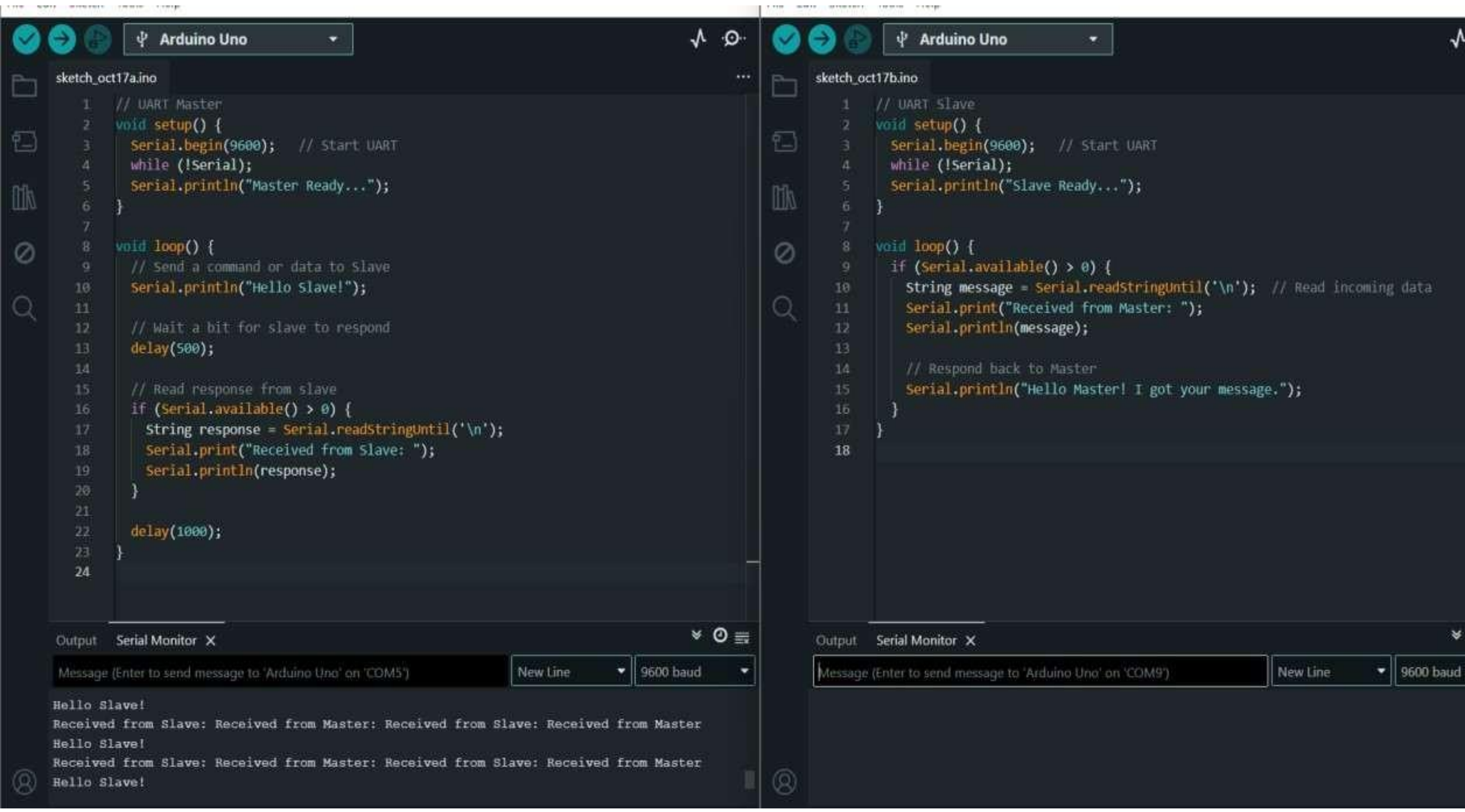
A key advantage of SPI is its full-duplex capability, which allows data to be sent and received at the same time. Its simplicity and high throughput make it an excellent choice for real-time embedded applications that require fast data exchange.



2. *Universal Asynchronous Receiver-Transmitter (UART)*

The Universal Asynchronous Receiver and Transmitter (UART) protocol facilitates serial communication without the need for a shared clock signal. Instead, devices agree on a specific data transmission speed, known as the baud rate. Communication occurs over two wires: TX (Transmit) for sending data and RX (Receive) for receiving it, enabling full-duplex data flow.

Data is transmitted in packets or "frames," which include a start bit, the actual data bits, an optional parity bit for error checking, and one or more stop bits to signal the end of the frame. In this lab, two Arduino boards were connected to demonstrate UART, with one board programmed to transmit data and the other to receive and display it on the Serial Monitor.

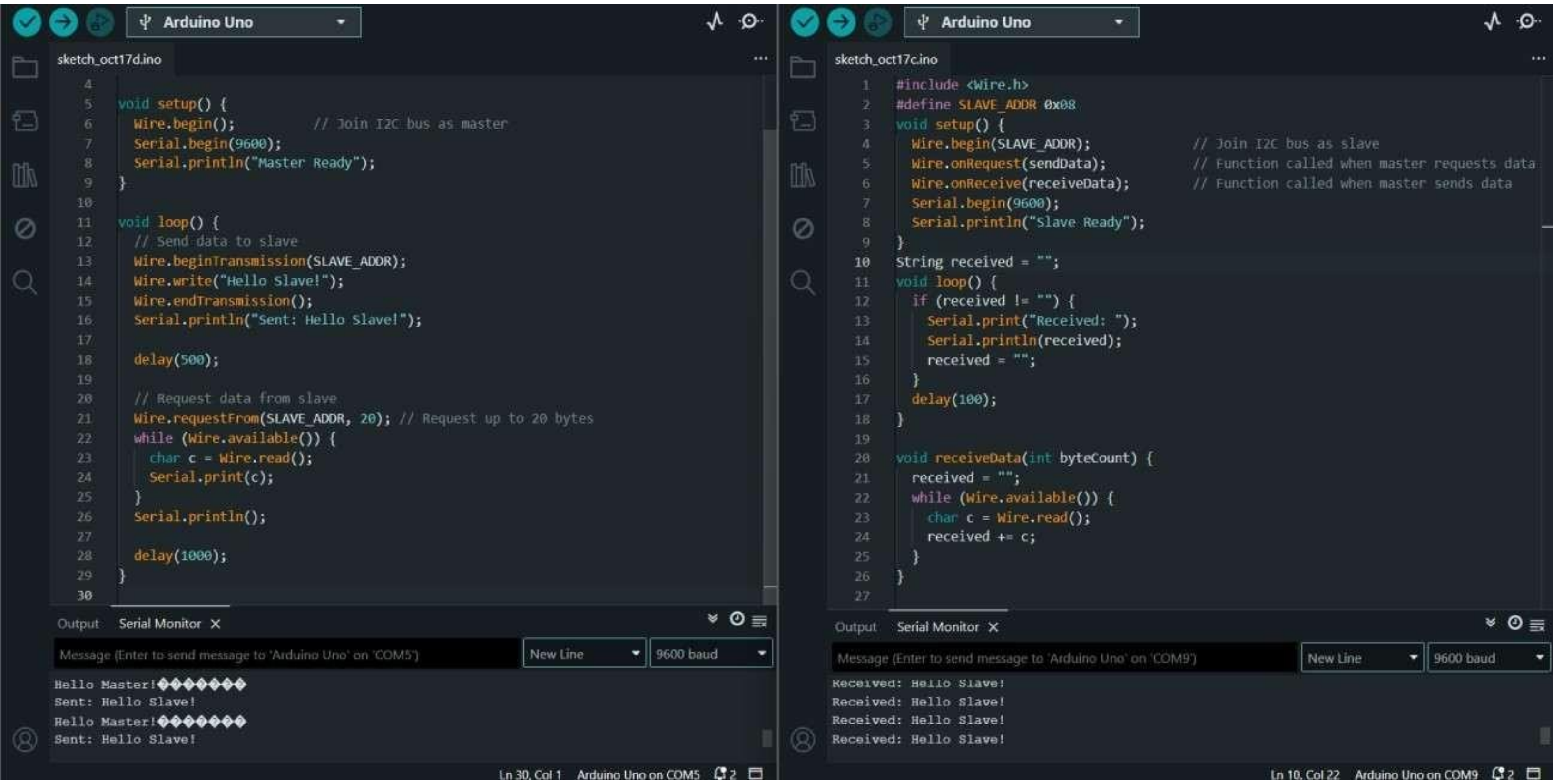


3. *Inter-Integrated Circuit (I2C)*

I²C (Inter-Integrated Circuit) is a popular serial protocol ideal for communication between microcontrollers and various peripherals over short distances. Its most notable feature is that it only requires two wires to create a bus that can be shared by multiple devices:

- SDA (Serial Data Line): Transmits the actual data.
- SCL (Serial Clock Line): Carries the synchronizing clock signal.

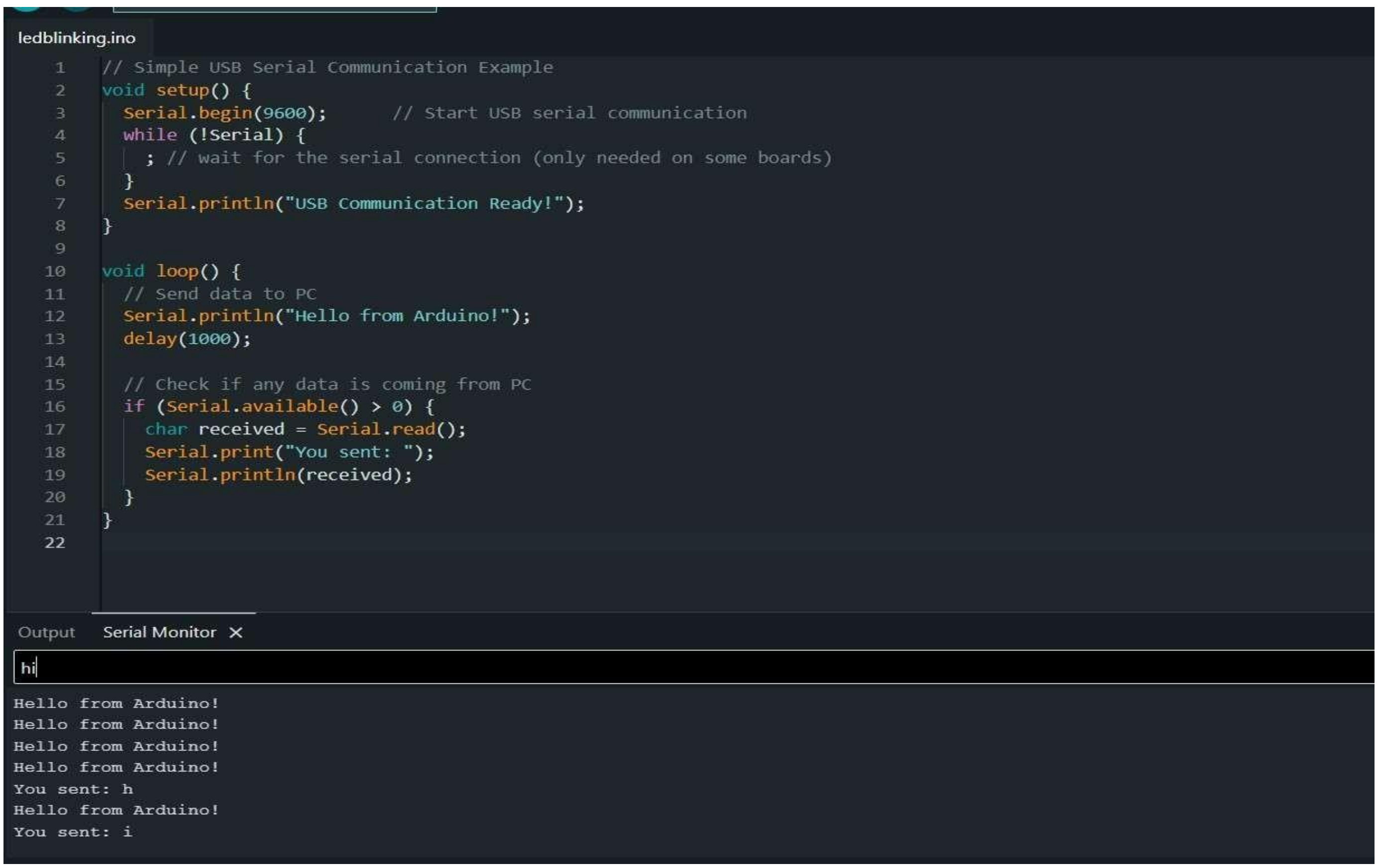
The protocol supports both multi-master and multi-slave configurations, where each device on the bus is assigned a unique 7-bit or 10-bit address. Communication is initiated when a master sends a start condition, followed by the slave's address. The receiving device confirms receipt of each byte with an acknowledgment (ACK) signal. While I²C is slower than SPI, its simple wiring makes it extremely efficient for connecting numerous devices to a single microcontroller.



4. *Universal Serial Bus (USB)*

The Universal Serial Bus (USB) is a standardized protocol designed to simplify the connection of peripheral devices—such as keyboards, mice, and storage drives—to computers. It is a host-controlled protocol, where the computer (the host) initiates and manages all communication with the connected peripherals (the devices).

Key features of USB include plug-and-play functionality, which allows devices to be automatically recognized and configured, and hot-swapping, which means devices can be connected or disconnected while the system is running. A standard USB cable contains four wires: two for power (VCC and GND) and two for differential data signals (D+ and D-). This versatile standard is widely used for its high-speed data transfer capabilities and its ability to deliver power to connected devices.



Result

The experiment successfully demonstrated the distinct characteristics of four primary serial communication protocols:

- SPI: Confirmed as a high-speed, synchronous, full-duplex protocol ideal for one-to-one or one-to-few connections where performance is critical. It uses four wires and a dedicated slave select line instead of addresses.
- I²C: Verified as a two-wire synchronous protocol that excels at connecting multiple slave devices to a single bus using a unique addressing scheme. It is simpler to wire than SPI for multi-device setups but operates at a lower speed.
- UART: Shown to be an effective asynchronous protocol for point-topoint, full-duplex communication. It relies on a pre-configured baud rate for synchronization and is commonly used for debugging and connecting modules like GPS or Bluetooth.
- USB: Implemented as a host-controlled protocol for connecting peripherals to a computer. Its architecture supports plug-and-play, power delivery, and high-speed, packet-based data transfer, making it a universal standard.

Conclusion

The experiment demonstrated and confirmed the functionality of USB, UART, I^2C, and SPI communication protocols using the Arduino platform. It showed that Arduino can effectively handle various serial communication tasks, including USB-based PC interfacing and communication with microcontrollers and peripherals via UART, I^2C, and SPI. The lab work highlights Arduino's suitability for embedded systems, whether for straightforward point-to-point connections or more complex multi-device networks. All four protocols were successfully tested using Arduino Uno and the Arduino IDE.