UART PROTOCOL

SWIPE

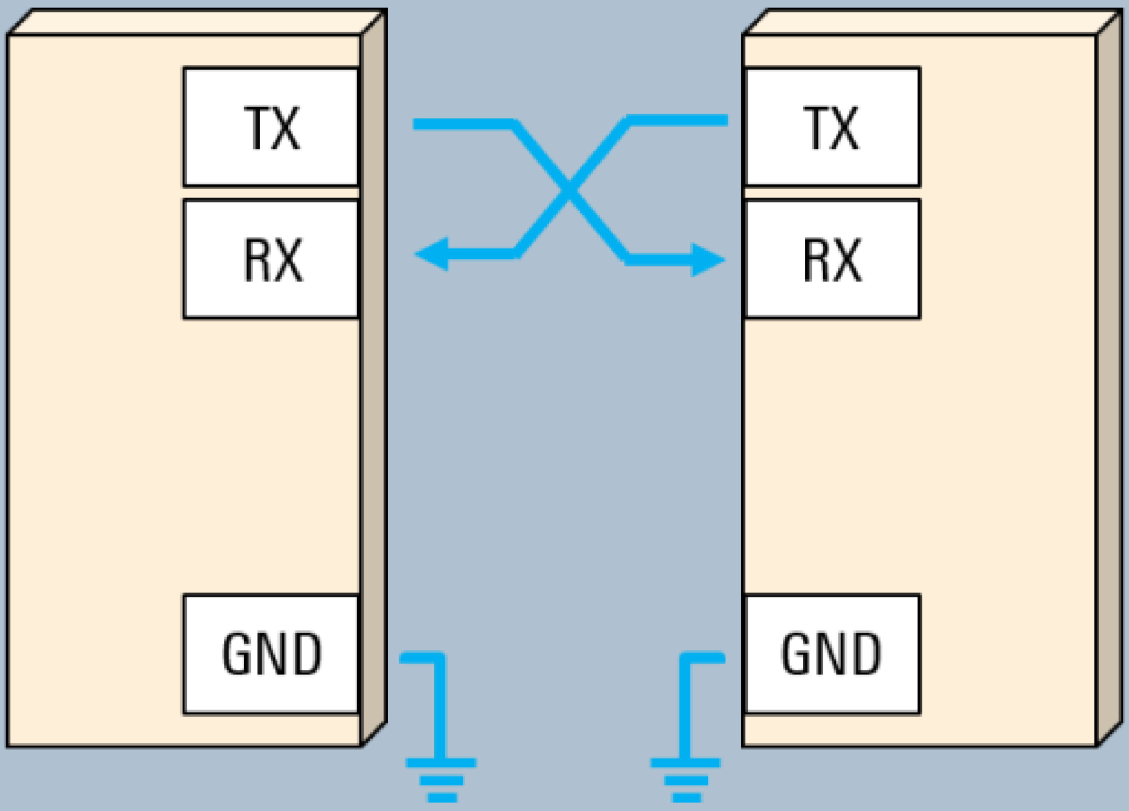
- UART stands for **universal asynchronous receiver / transmitter.**

- Exchange **serial data** between two devices.

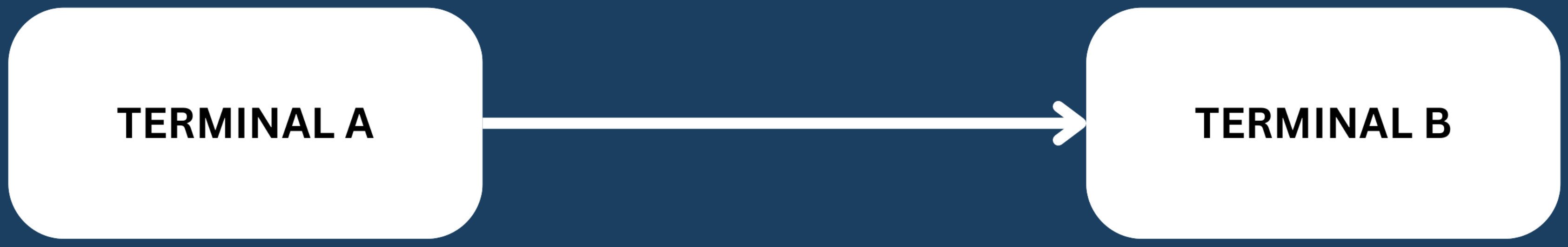
- It uses **two wires** between transmitter and receiver to transmit and receive in both directions

- Communication in UART can be

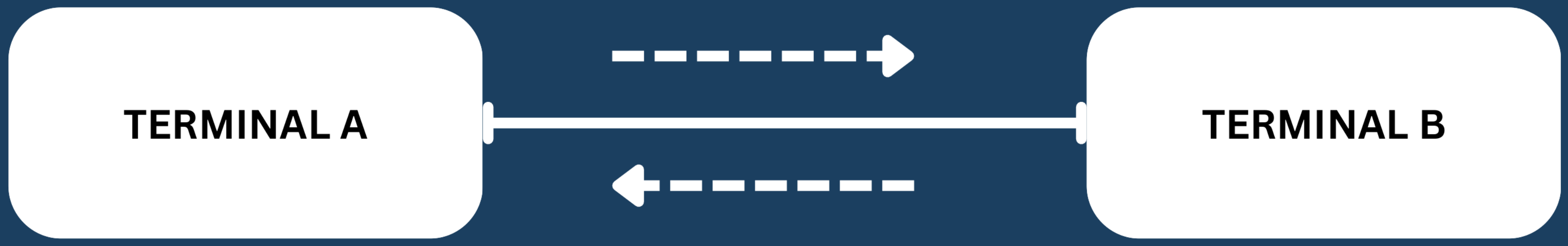
1. **Simplex -** Transmission in one direction.
2. **Half-duplex -** Transmission in either direction but not simultaneously .
3. **Full-duplex-** Transmission in both directions simultaneously.



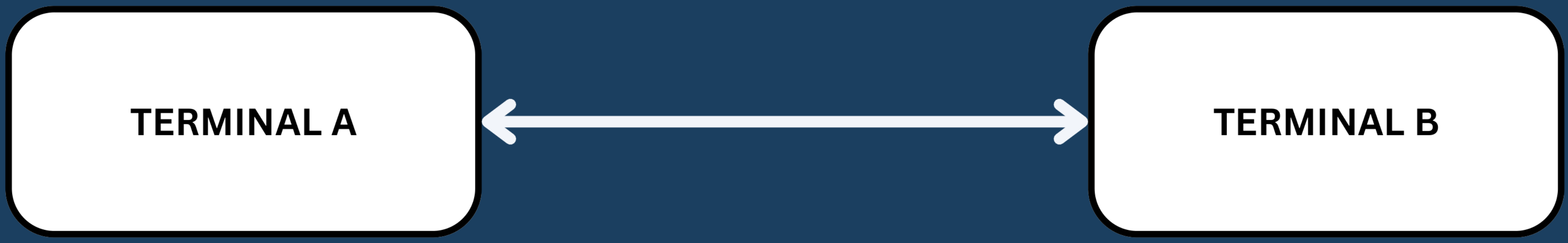
1.Simplex



2.Half-duplex



3.Full-duplex



ABOUT UART

- UART is still used for **lower-speed and lower-throughput applications**, because it is very simple, low-cost and easy to implement.

- One of the big advantages of UART is that it is asynchronous – the transmitter and receiver do not share a common clock signal.

- Since they do not share a clock, both ends must transmit at the same and has same baud rate.

- The most common **UART baud rates** in use today are 4800, 9600, 19.2K, 57.6K, and 115.2K.

- The popularity of UART has decreased: protocols like SPI and I2C have been replacing UART.

UART frame format

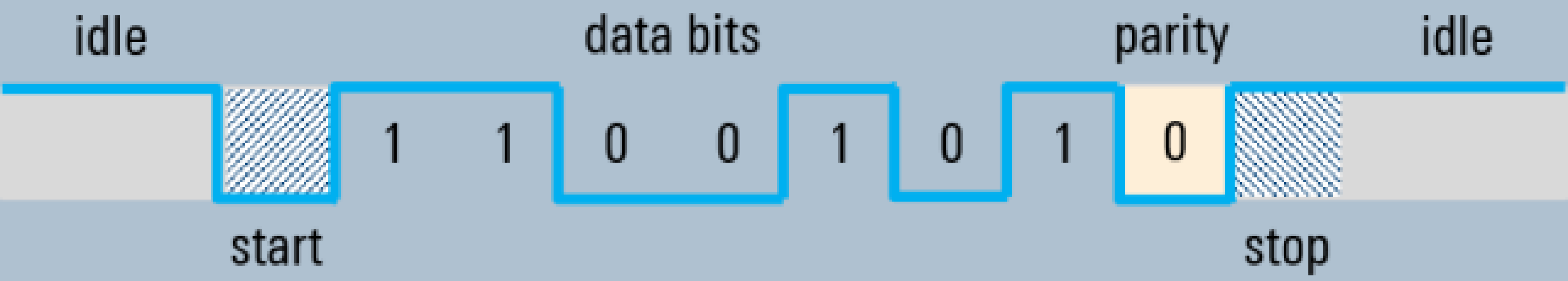
- Data in UART is transmitted in the form of frames.

- UART frames contain start and stop bits, data bits and optional parity bit.

- Normally,

1. HIGH voltage -> 1
2. LOW voltage-> 0

- Since the UART protocol doesn’t define specific voltages or voltage ranges for these levels, sometimes high is also called “mark” while low is called “space”.



- The start bit is a transition from the idle high state to a low state.

- The stop bit is either a transition back to the high or idle state or remaining at the high state for an additional bit time.

- There can be 5 to 9 user data bits, although 7 or 8 bits is most common. These data bits are usually transmitted with the least significant bit first.

Example:

- If we want to send the capital letter “S” in 7-bit ASCII, the bit sequence is 1 0 1 0 0 1 1. We first reverse the order of the bits to put them in least significant bit order, that is 1 1 0 0 1 0 1, before sending them out. After the last data bit is sent, the stop bit is used to end the frame and the line returns to the idle state.

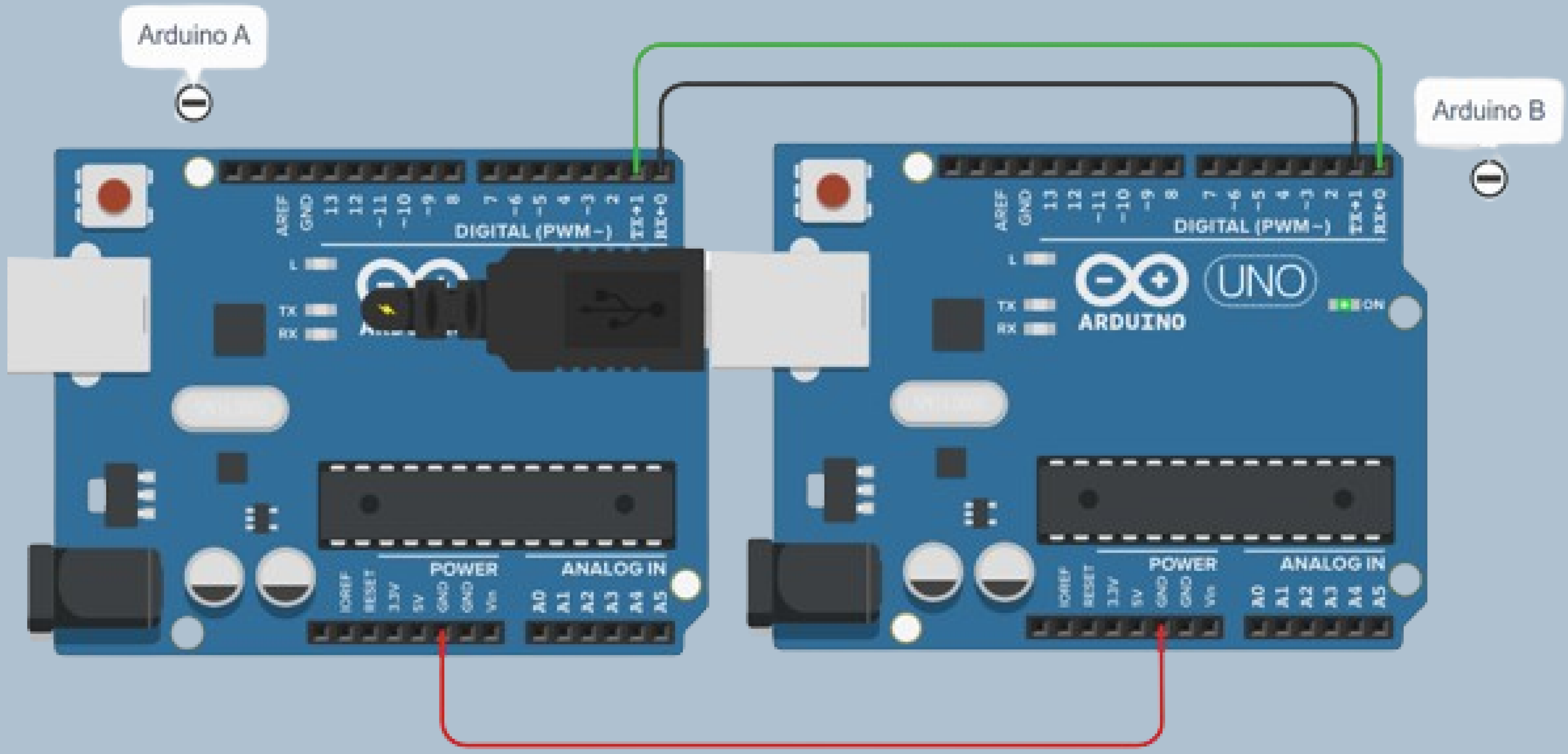
- 7-bit ASCII ‘S’ (0x52) = 1 0 1 0 0 1 1
- LSB order = 1 1 0 0 1 0 1

PARITY BIT

- A UART frame can also contain an optional parity bit that can be used for error detection.
- This bit is inserted between the end of the data bits and the stop bit. The value of the parity bit depends on the type of parity being used (even or odd):
- In even parity, this bit is set such that the total number of 1s in the frame will be even.
- In odd parity, this bit is set such that the total number of 1s in the frame will be odd.

Example

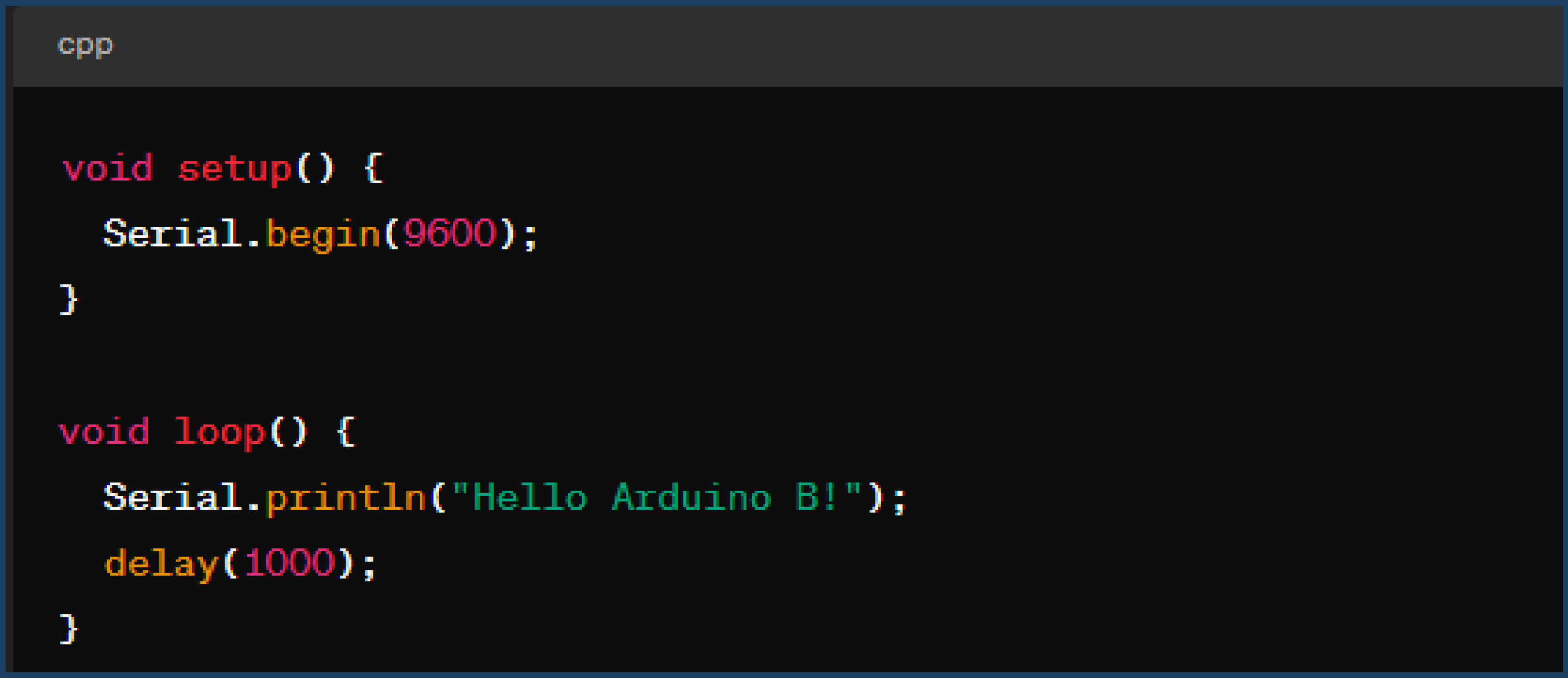
Lets demonstarte a UART communication in arduino uno



Remember to connect the RX pin of one Arduino to the TX pin of the other and vice versa. Also, make sure both Arduinos share a common ground.

1.Simplex

Arduino A (Transmitter):

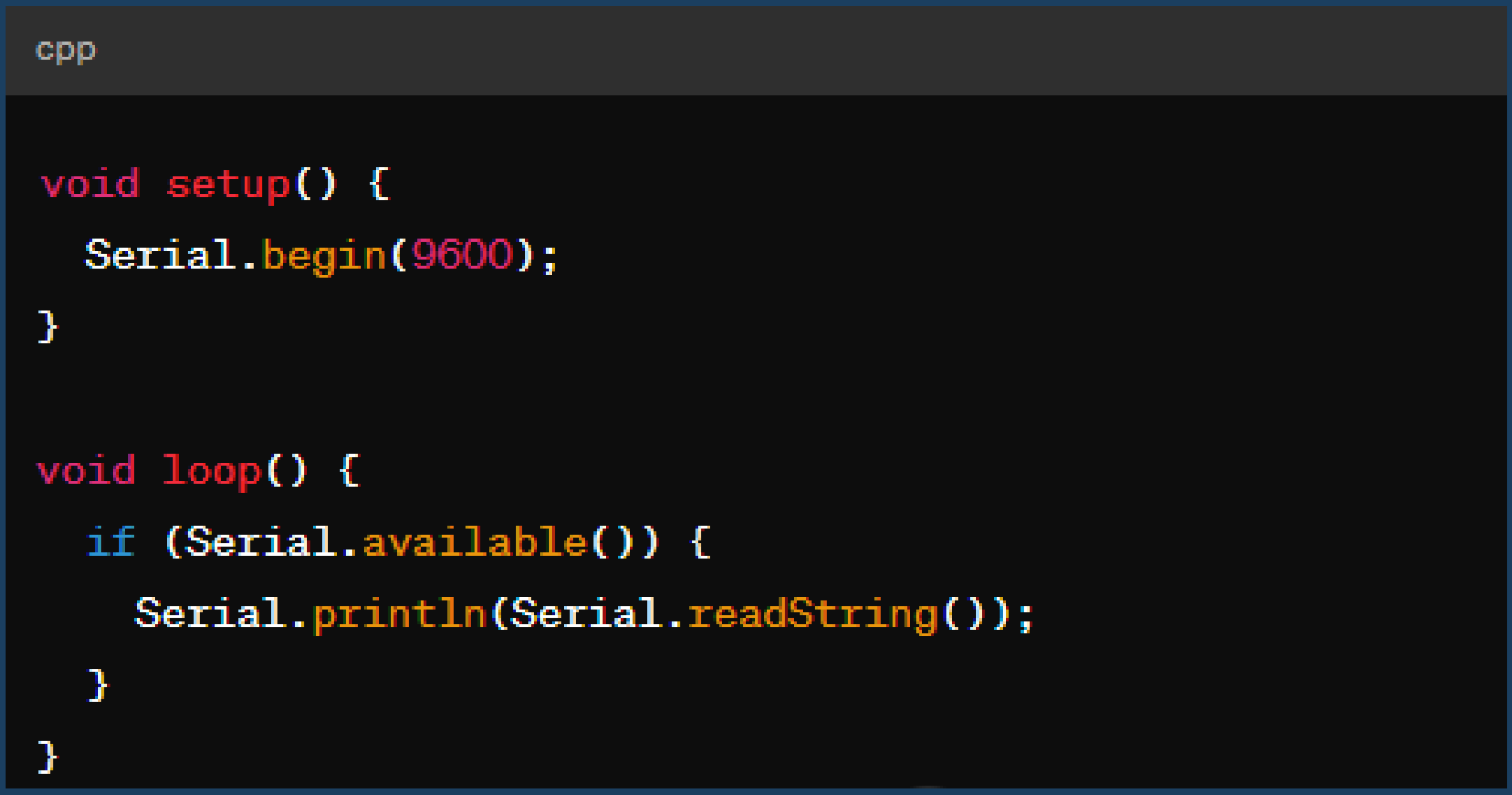


cpp

```cpp
void setup() {
  Serial.begin(9600);
}

void loop() {
  Serial.println("Hello Arduino B!");
  delay(1000);
}
```

Arduino B (Receiver):



cpp

```cpp
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    Serial.println(Serial.readString());
  }
}
```

2.Half-duplex

Arduino A :

```cpp
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    Serial.println(Serial.readString());
  }
  delay(1000);
  Serial.println("Hello Arduino B!");
}
```

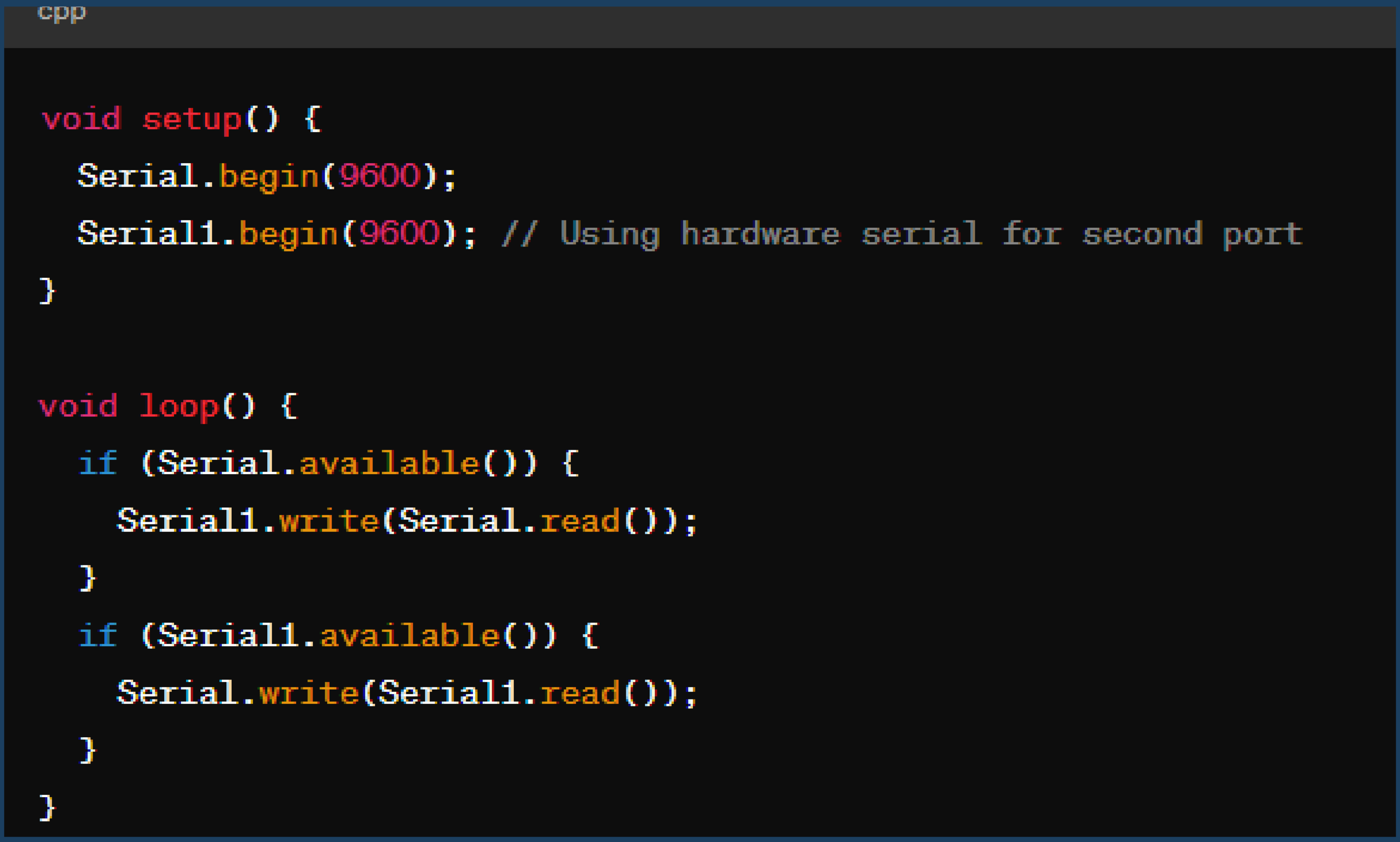
Arduino B :

```cpp
void setup() {
  Serial.begin(9600);
}

void loop() {
  if (Serial.available()) {
    Serial.println(Serial.readString());
  }
  delay(1000);
  Serial.println("Hello Arduino A!");
}
```

3.Full-duplex

Arduino A :

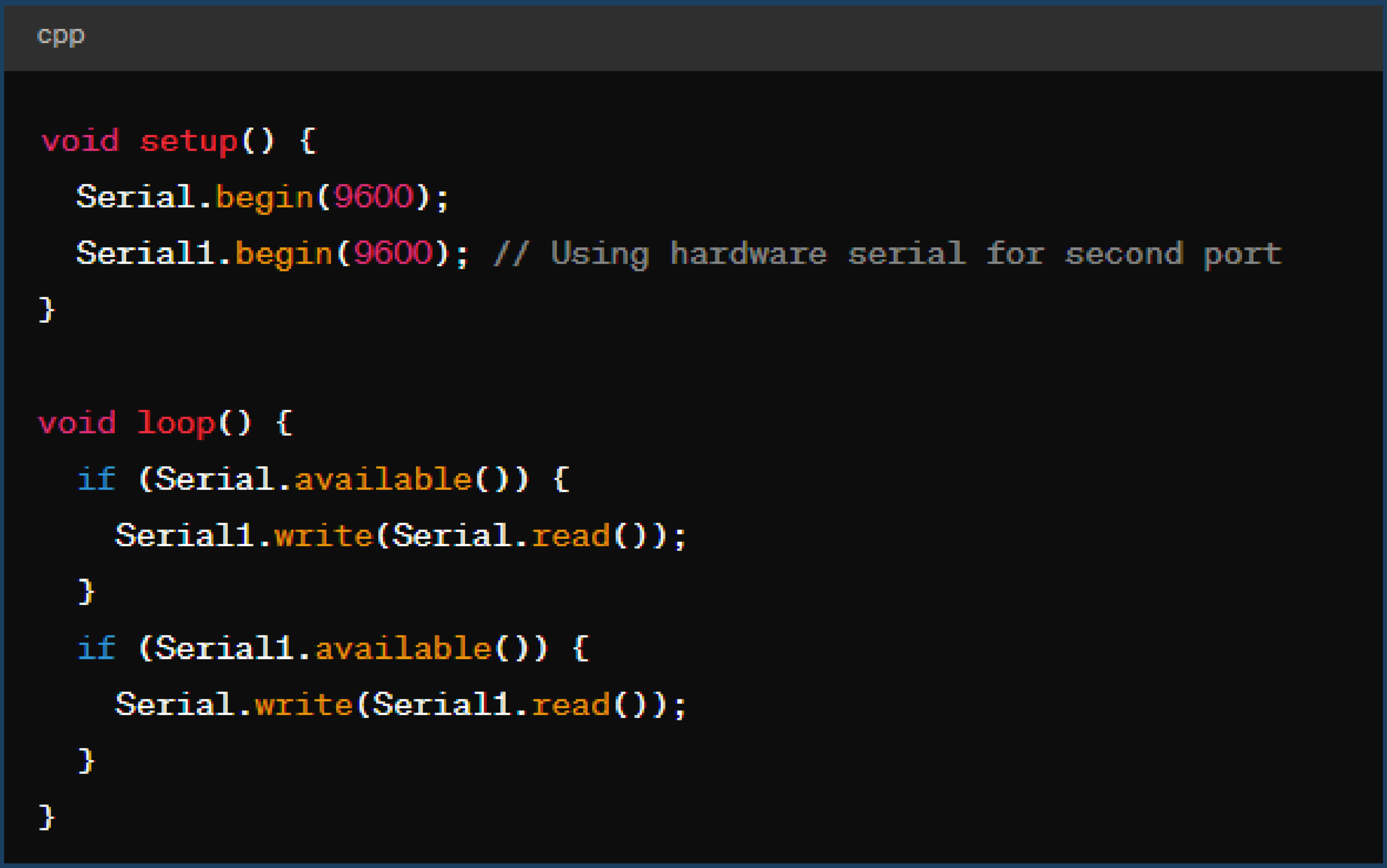


cpp

```cpp
void setup() {
  Serial.begin(9600);
  Serial1.begin(9600); // Using hardware serial for second port
}

void loop() {
  if (Serial.available()) {
    Serial1.write(Serial.read());
  }
  if (Serial1.available()) {
    Serial.write(Serial1.read());
  }
}
```

Arduino B :



cpp

```cpp
void setup() {
  Serial.begin(9600);
  Serial1.begin(9600); // Using hardware serial for second port
}

void loop() {
  if (Serial.available()) {
    Serial1.write(Serial.read());
  }
  if (Serial1.available()) {
    Serial.write(Serial1.read());
  }
}
```