Faculty of Engineering and Technology
Department of Electrical and Computer Engineering

ENCS 4110
COMPUTER DESIGN LABORATORY
Report #3
Experiment #3

"Serial Communication with Arduino"

Prepared by:

Nsreen Tawafsha – 1182319

Supervised by:

Dr. Ahmad Afaneh

Teacher Assistant:

Eng. Heba Qadah

Section: 4

Date: 11 March 2021

1. Abstract

The aim of this experiment is to learn and apply the fundamental concepts of serial communication, as well as how the Arduino board responds to them.

Table of Content

List of Figures

List of Tables

2. Theory

2.1 Arduino:-

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing. As an example, the pins of an Arduino board called Arduino Uno (R3) are shown in **Figure 2.1.1**.

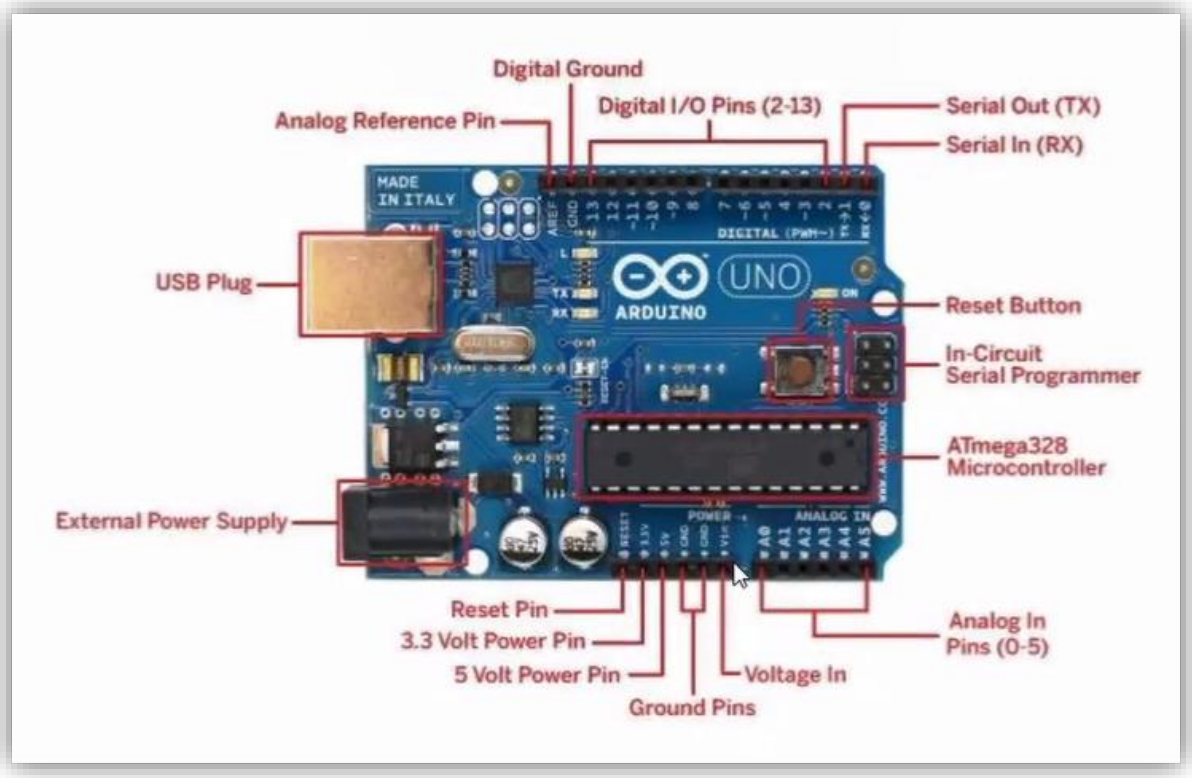

Figure 2.1.1: Arduino Uno (R3) construction

2.2 UART:-

Universal Asynchronous Receiver/Transmitter (UART) is an acronym for Universal Asynchronous Receiver/Transmitter. It is a physical circuit in a microcontroller or a stand-alone IC, not a communication protocol like SPI or I2C. The primary feature of a UART is to send and receive serial data. **Figure 2.2.1** shown this circuit.

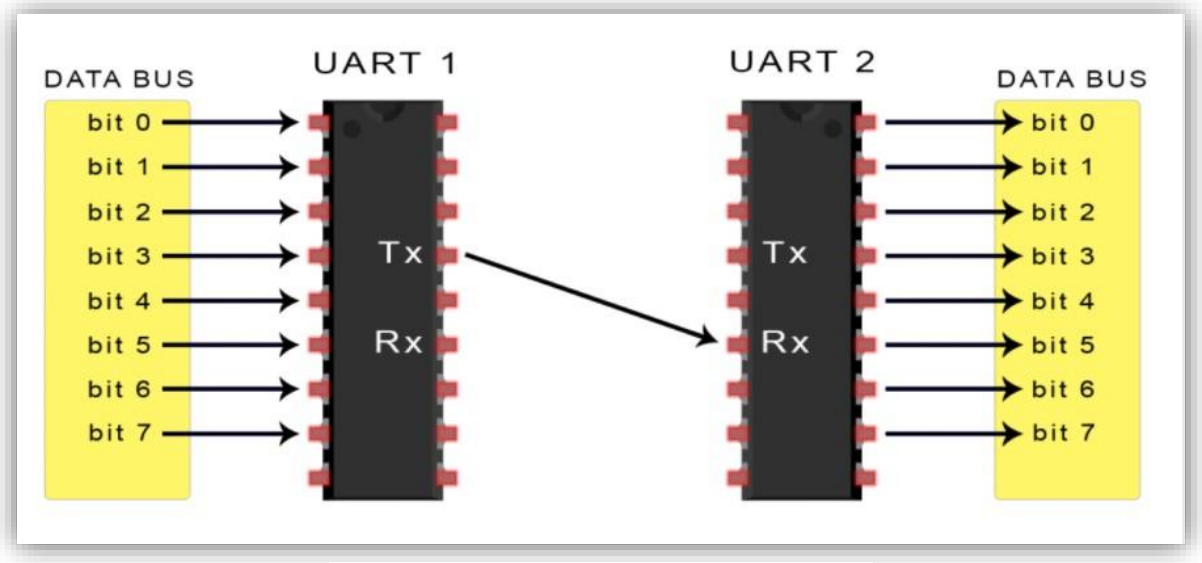

Figure 2.2.1: UART circuit construction

UARTs transmit data asynchronously, which means there is no clock signal to synchronize the transmitting UART's output of bits with the receiving UART's sampling of bits. The transmitting UART adds start and stop bits to the data packet being transmitted instead of a clock signal. When a start bit is detected by the receiving UART, it begins reading the incoming bits at a fixed frequency known as the baud rate. The baud rate is a unit of measurement for data transmission speed, expressed in bits per second (bps). Both UARTs must communicate at a similar baud rate. The baud rate difference between the transmitting and receiving UARTs can only be around 10% before the bit timing gets out of hand.

UART transmitted data is organized into packets. Each packet contains 1 start bit, 5 to 9 data bits (depending on the UART), an optional parity bit, and 1 or 2 stop bits. **Figure 2.2.2** illustrates this representation.

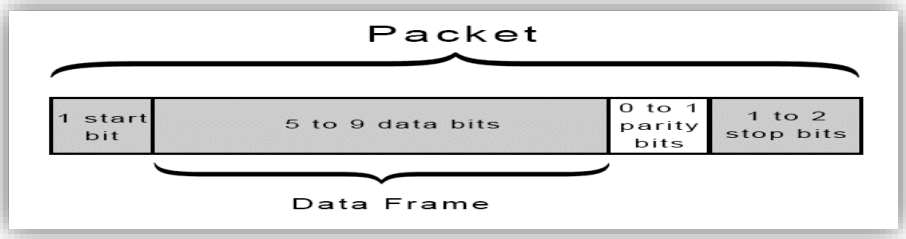

Figure 2.2.2: UART packet of data representation

2.3 Serial communication with Arduino [Tx, Rx]:-

Serial communications provide an easy and flexible way for the Arduino board to interact with any computer and other devices. All Arduino boards have at least one serial port (also known as a UART or USART), and some have several.

You can use the Arduino environment's built-in serial monitor to communicate with an Arduino board. Click the serial monitor button in the toolbar and select the same baud rate used in the call to begin().

Serial communication on pins TX/RX uses TTL logic levels (5V or 3.3V depending on the board). Don't connect these pins directly to an RS232 serial port; they operate at +/- 12V and can damage your Arduino board.

Table 2.3.1 illustrates some serial communication finctions.

Table 2.3.1: Serial communication functions

Function	***explanation***
Serial.begin(speed) Serial.begin(speed, config)	Sets the data rate in bits per second (baud) for serial data transmission.
Serial.available()	Get the number of bytes (characters) available for reading from the serial port.
Serial.read()	Reads incoming serial data.
Serial.print()	Prints data to the serial port as human-readable ASCII text.
Serial.readBytesUntil(character, buffer, length)	reads characters from the serial buffer into an array.
Serial.write()	Writes binary data to the serial port
Serial.parseInt()	Looks for the next valid integer in the incoming serial.

3. Procedure and Discussion

3.1 Part1: Basic communication between Arduino & PC

The idea of this part is sending a simple information between the Arduino & PC ["Nsreen Tawafsha"] in order to become familiar with the connections & the setup. **Figure 3.1.1** shows the circuit, code and the simulation of this part.

Required Components:

1. PC
2. Arduino UNO Board
3. USB Cable

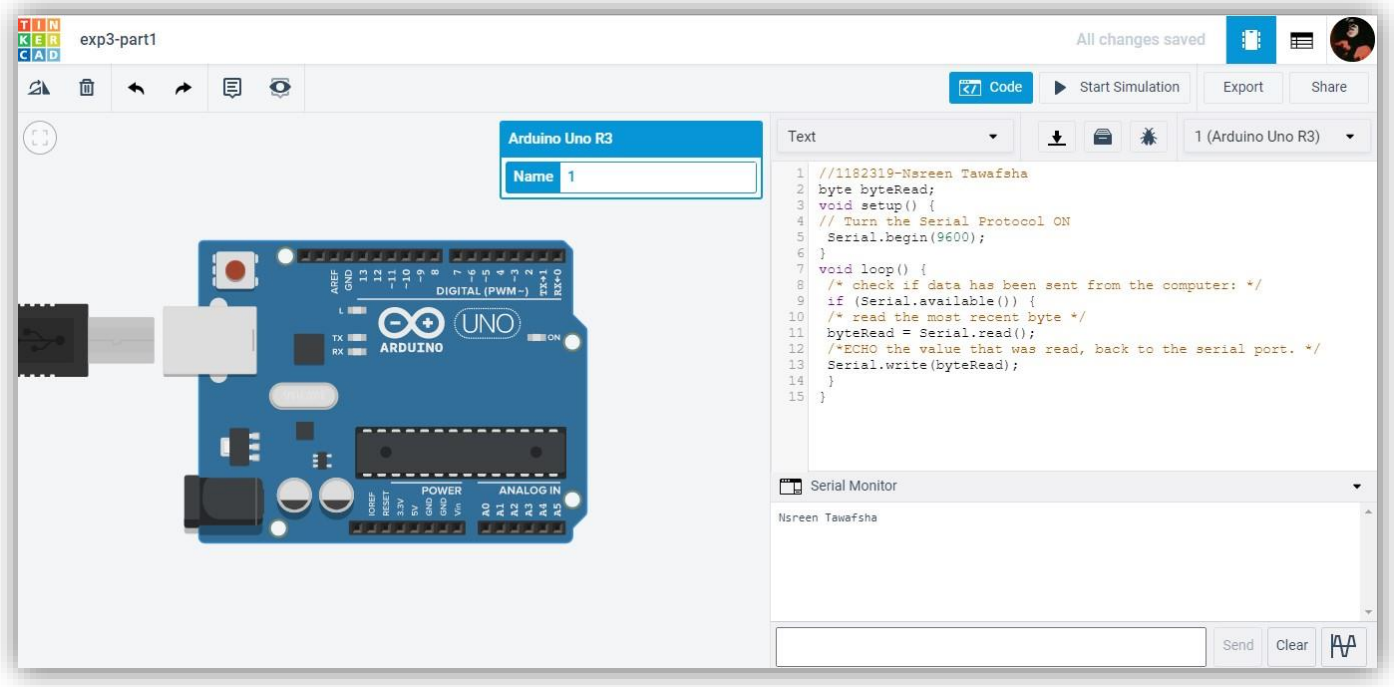

Figure 3.1.1: Basic communication between Arduino & PC circuit

This code is used to read the sentence taken by the user using Serial.read() function, then print it at serial port using Serial.write() function as shown in **Figure 3.1.1**

3.2 Part2: Basic communication between 2 Arduinos

The idea of this part is sending a simple information between the 2 Arduinos in order to become familiar with the connections & the setup.

In this part, TX in the First Arduino must connected with RX in the second one and Vice versa [TX1 ↔ RX2 & RX1 ↔ TX2]. The ground must be common between both of them. **Figure 3.2.1** shows the representation of this circuit.

Required Components:

1. Two Arduino UNO Boards
2. Jumper Wires

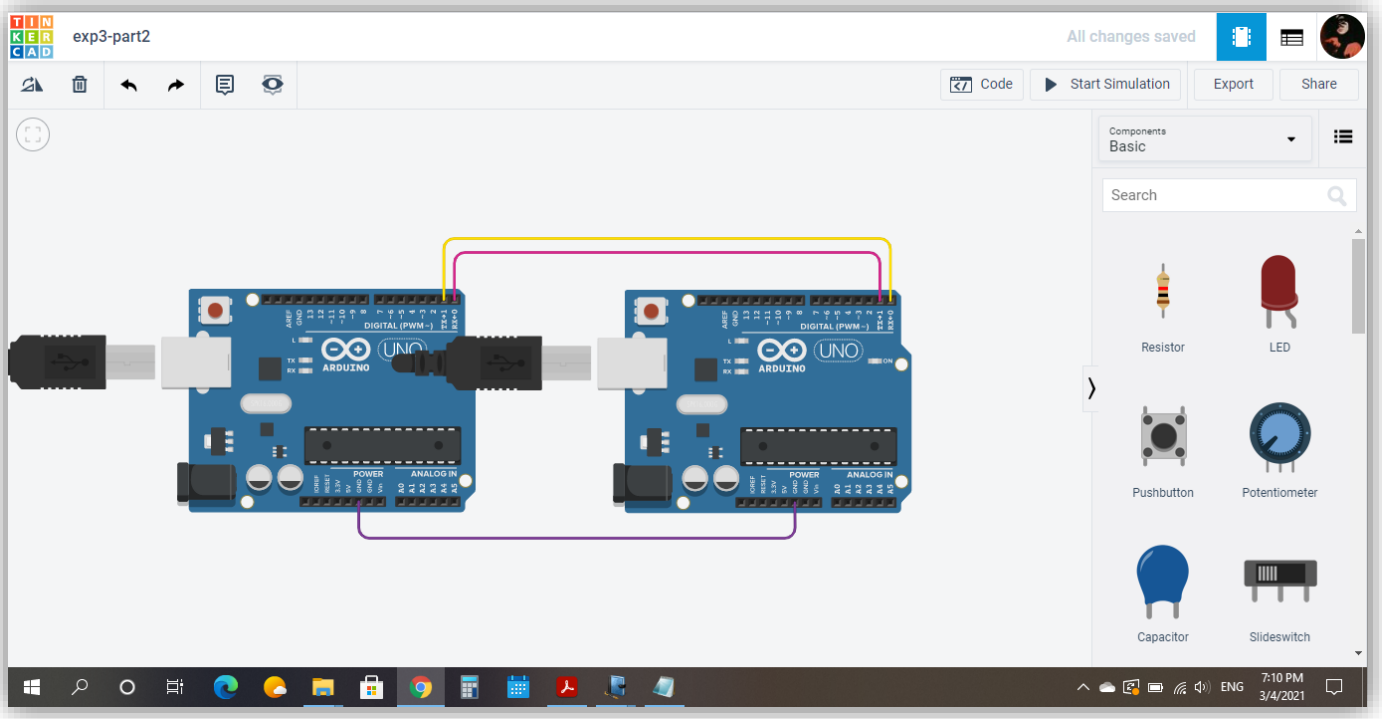

Figure 3.2.1: Communication between 2 Arduinos circuit

After that, writing the code shows on **Figure 3.2.2** for the first Arduino (Sender one) and the code shows on **Figure 3.2.3** for the second (Receiver one) then run it:

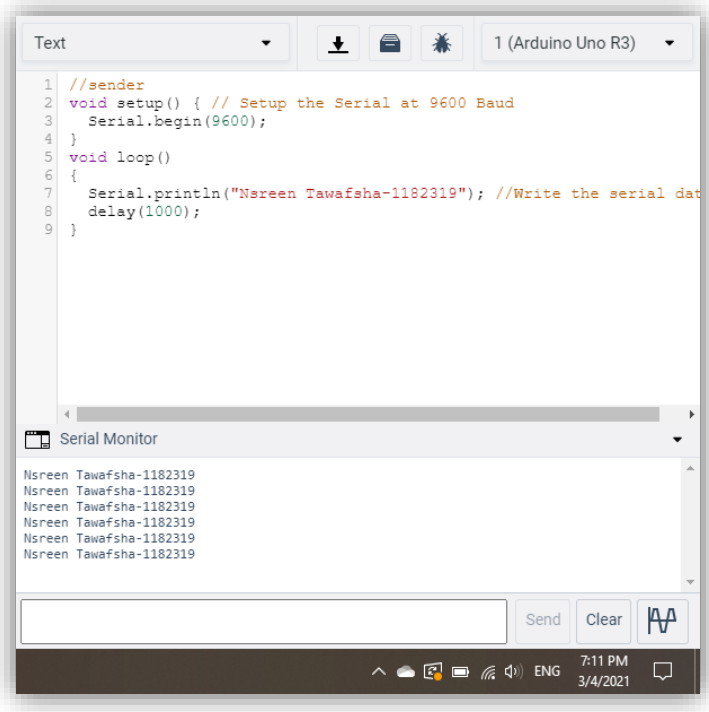

Figure 3.2.2: Sender Arduino Code

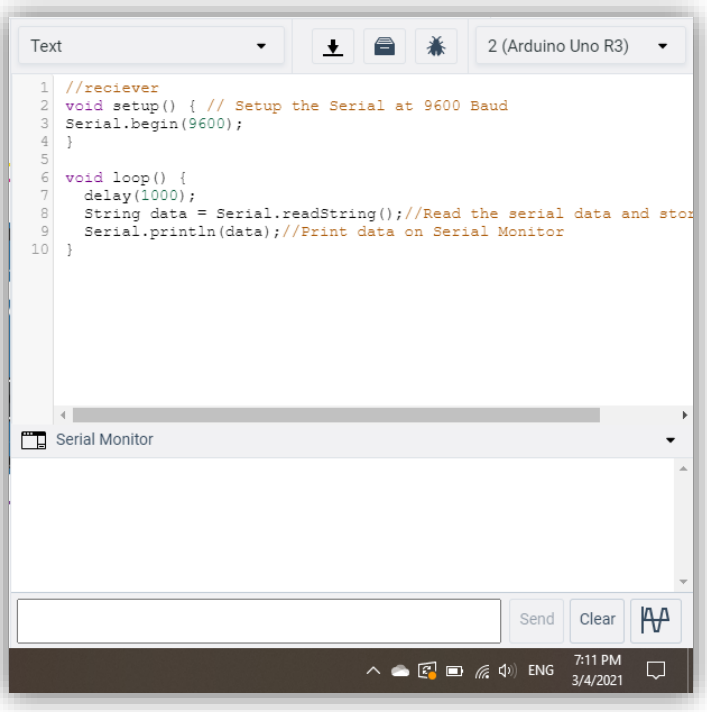

Figure 3.2.3: Receiver Arduino Code

The sender Arduino code is used to print the sentence “Nsreen Tawafsha-1182319” using Serial.println() function at a loop on the receiver Arduino after specify the serial communication Setup using Serial.begin(9600). The receiver Arduino will read this sentence as appearing in the receiver code above using Serial.readString() function then prints data (the sentence received) to the serial port as human-readable ASCII text as shown in **Figure 3.2.4**

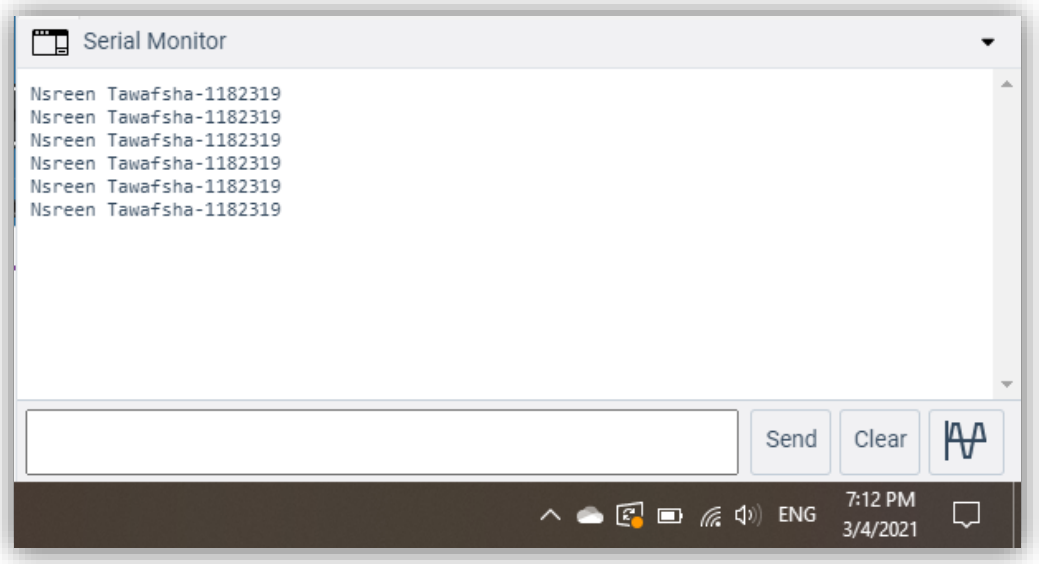

Figure 3.2.4: Communication between 2 Arduinos simulation

3.3 Part3: Push Button & LED using 2 Arduinos

The idea of this part is to connect a push button with Arduino1 and whenever it's pushed, the value 1 is sent to the Arduino2. In its part, Arduino2 receives the data and whenever logic 1 is read, a LED will be turned on. **Figure 3.3.1** shows the circuit of this part.

In this part, TX in the First Arduino must connected with RX in the second one and Vice versa [TX1 ↔ RX2 & RX1 ↔ TX2]. The ground must be common between both of them.

Required Components:

4. Two Arduino UNO Boards
5. Push Button
6. LED
7. Two Resistances ()

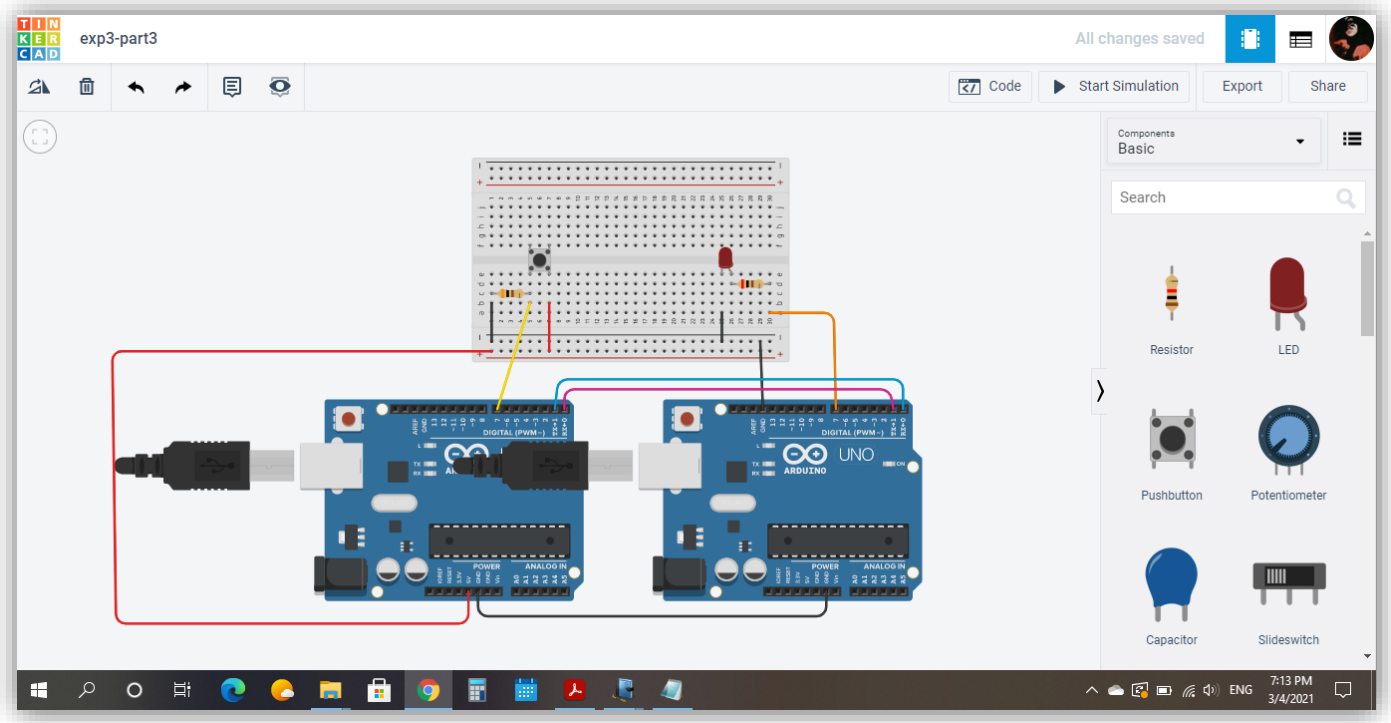

Figure 3.3.1: Push Button & LED using 2 Arduinos circuit

After that, writing the code shows on **Figure 3.3.2** for the first Arduino (Sender one) and the code shows on **Figure 3.3.3** for the second (Receiver one) then run it:

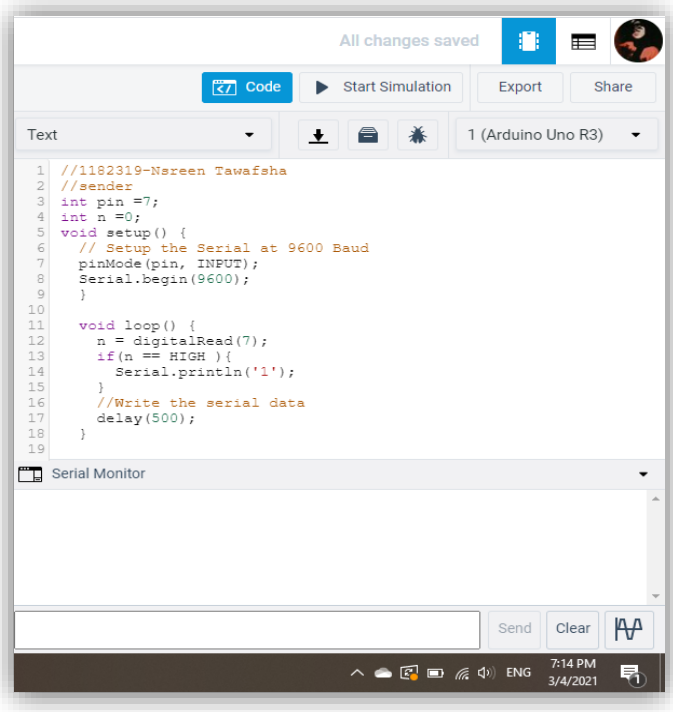

Figure 3.3.2: Sender Arduino Code

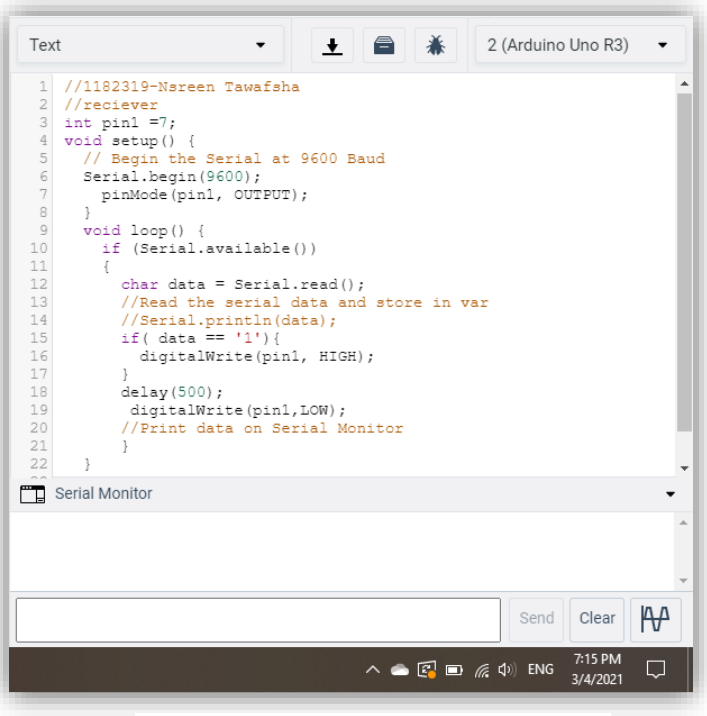

Figure 3.3.3: Receiver Arduino Code

The sender Arduino code is used to send the character '1' using Serial.println() function whenever the push button is pushed on the receiver Arduino after specify the serial communication Setup using Serial.begin(9600). If the push button not pushed, nothing should be sent. The receiver Arduino will read this character as appearing in the receiver code above using Serial.read() function then prints data (the character received) to the serial port as human-readable ASCII text and the Led must be turned on in this case as shown in **Figure 3.3.4**

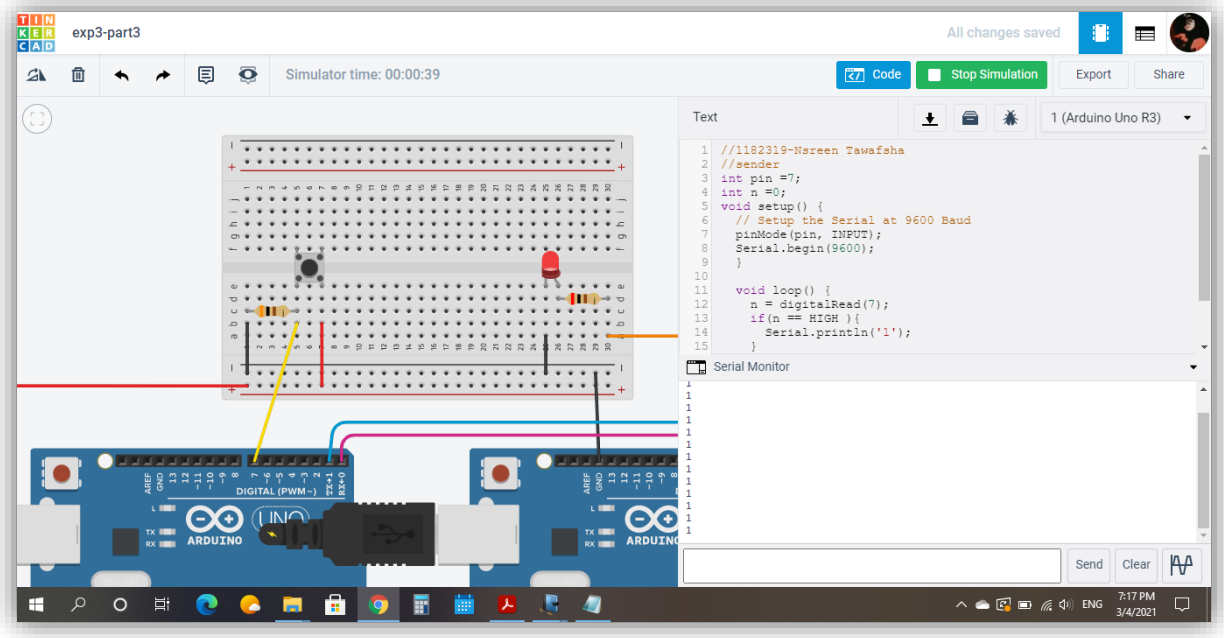

Figure 3.3.4: Push Button & LED using 2 Arduinos simulation

3.4 Part4: Visualization of serial communication using Serial Plotter

The idea of this part is to see how the serial data are transferred as start-data-parity-stop bits. This can be done using Serial plotter in Arduino IDE. **Figure 3.4.1** shows the circuit of this part.

Required Components:

1. Two Arduino UNO Boards
2. PC
3. USB Cable
4. Wires

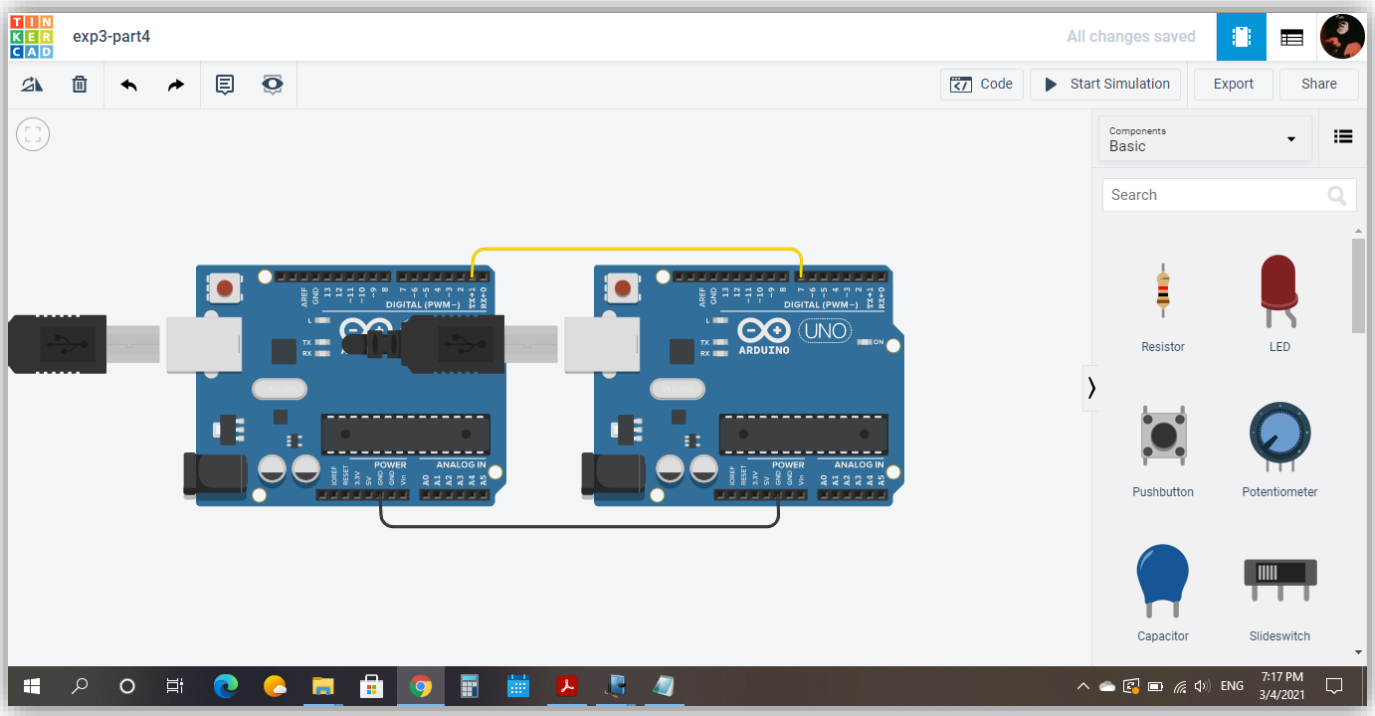

Figure 3.4.1: Visualization of serial communication using Serial Plotter circuit

After that, writing the code shows on **Figure 3.4.2** for the first Arduino (Sender one) and the code shows on **Figure 3.4.3** for the second (Receiver one) then run it:

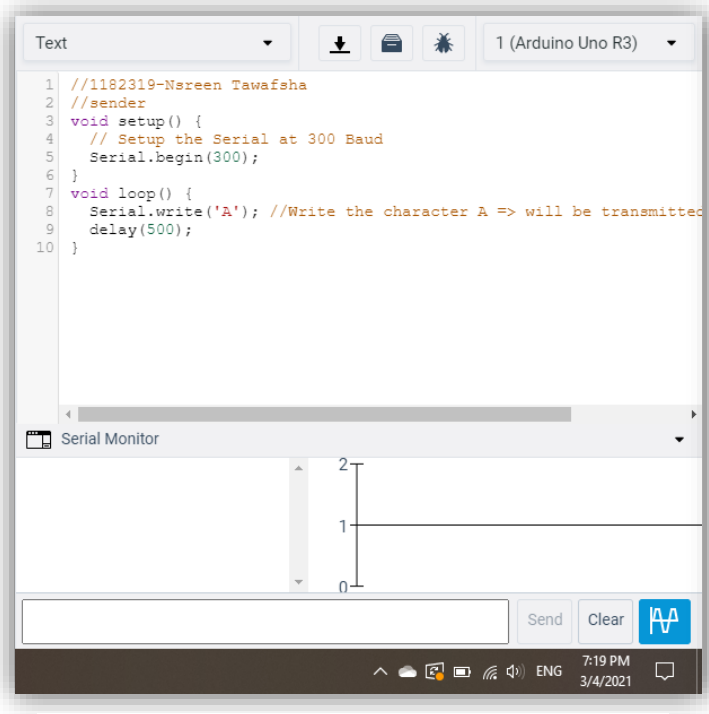

Figure 3.4.2: Sender Arduino Code

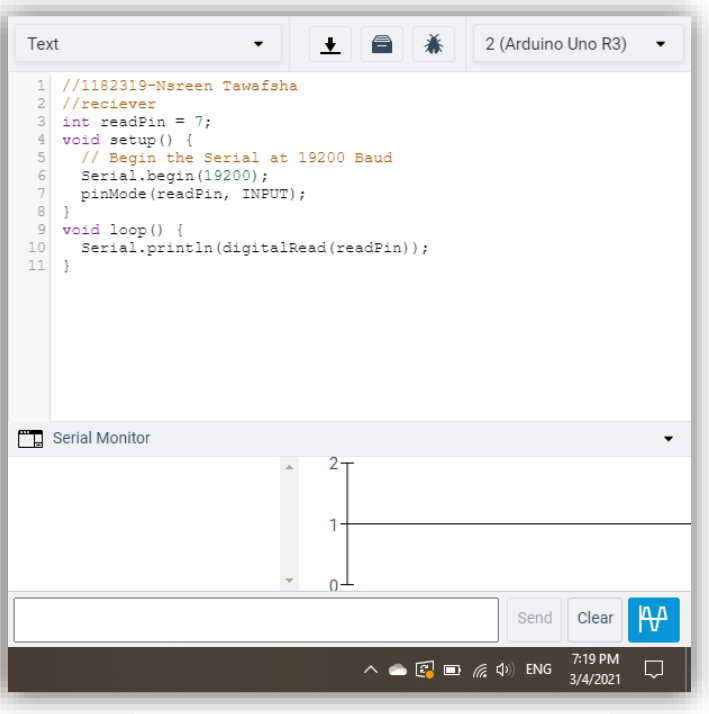

Figure 3.4.3: Receiver Arduino Code

The sender Arduino code is used to send the character 'A' to the receiver Arduino using Serial.write() function after setup the Baud Rate to be very small using Serial.begin(300) and it will write the character to the serial port as shown in **Figure 3.4.4**.

The receiver Arduino will read this character as appearing in the receiver code above using digitalRead with one of the digital pins (7 in the code) in order to plot the serial data as bits and print the bit caught to the serial port, a high Baud rate was used in the receiver Arduino using Serial.begin(19200) in order to capture the bits as shown in **Figure 3.4.5**

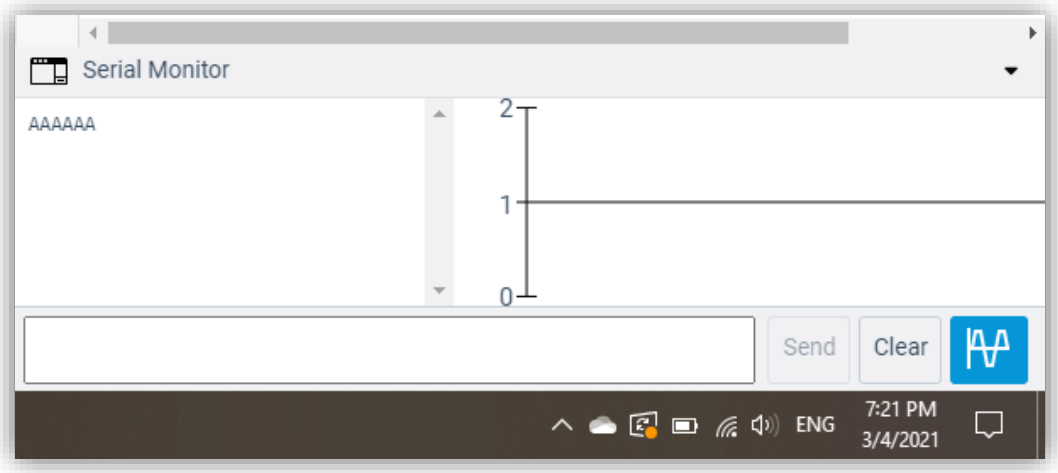

Figure 3.4.4: Sender Arduino code simulation

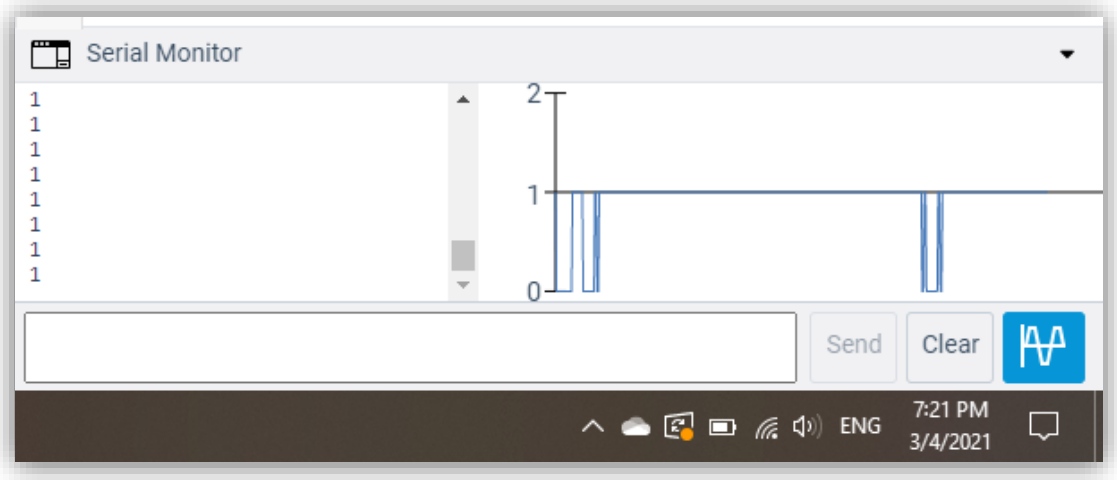

Figure 3.4.5: Receiver Arduino code simulation

3.5 Part5: Tasks

✓ **Task1: Draw the theoretical waveform for the start-data-parity-stop bits and compare it with what is plotted.**

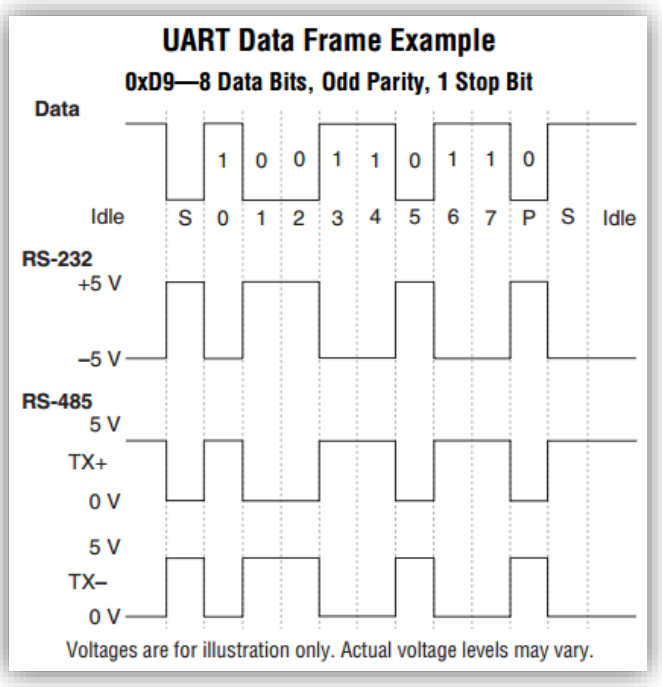

Figure 3.5.1: theoretical waveform

✓ **Task2: Try Sending different data. [Read the data from the user]**

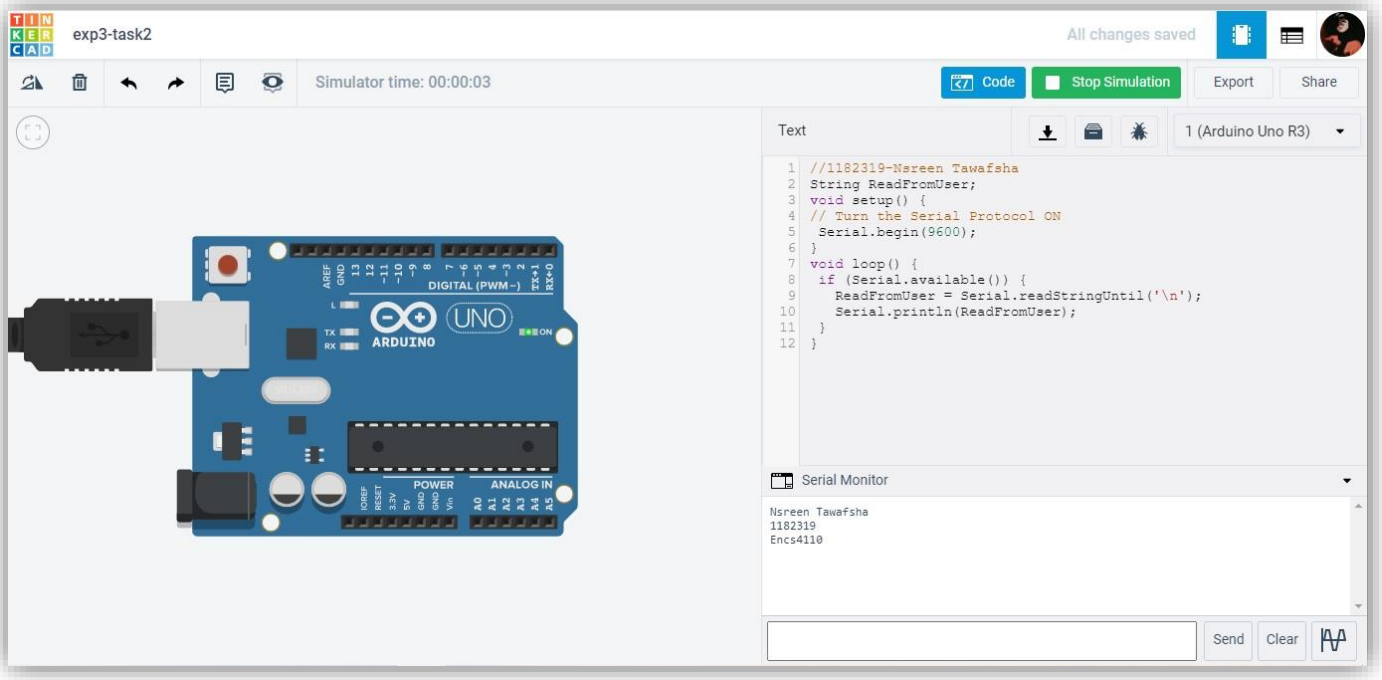

Figure 3.5.2: Read the data from the user

✓ **Task3: Try changing the serial setup:**

- **Even Parity**

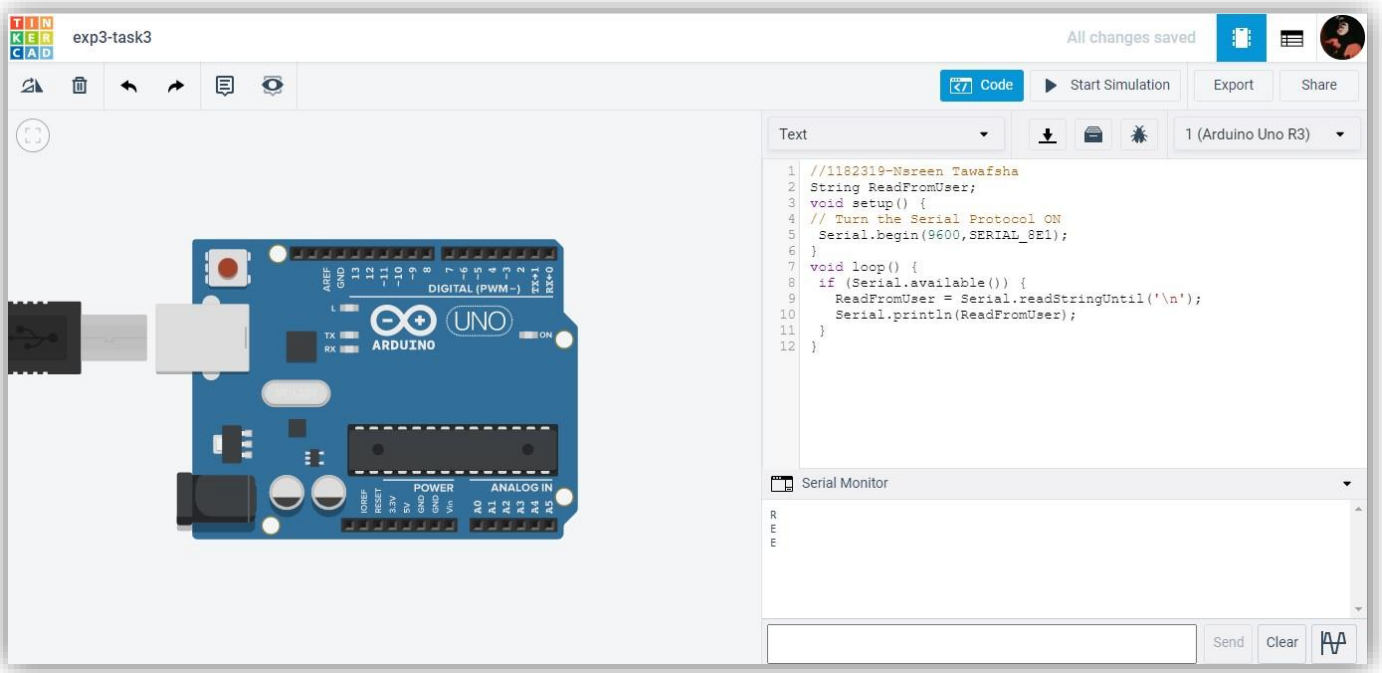

Figure 3.5.3: Even Parity

- **Odd Parity**

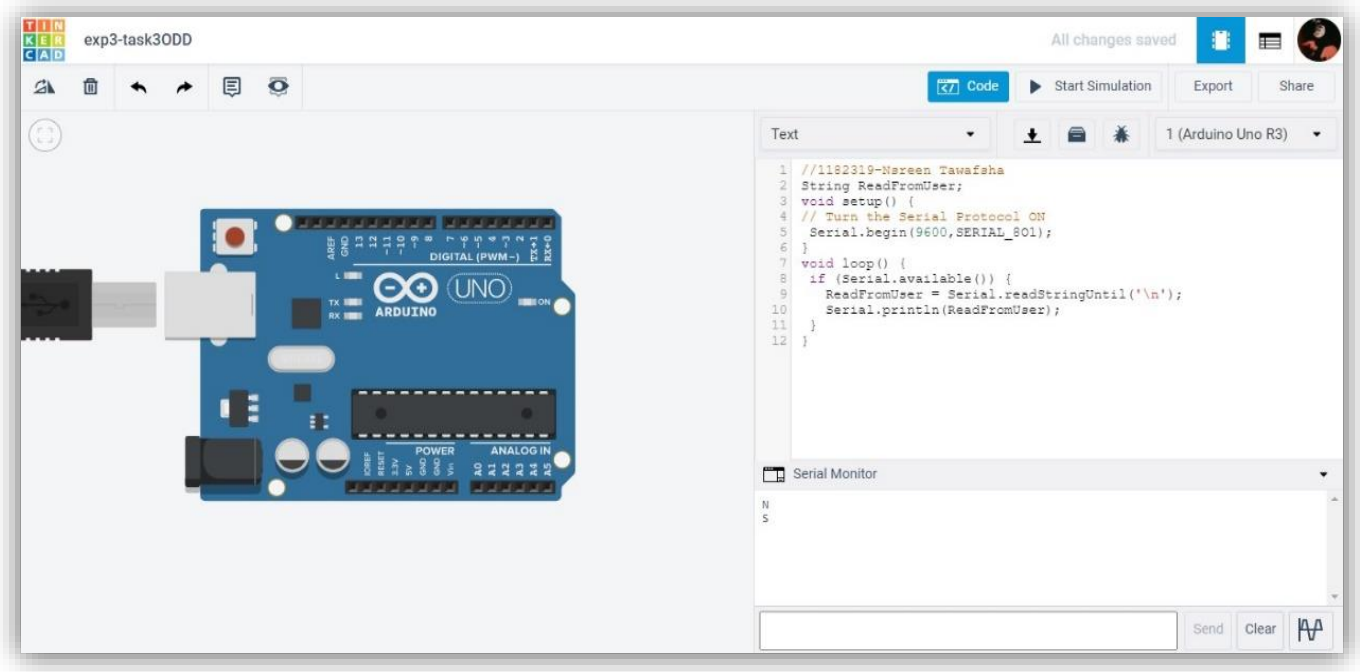

Figure 3.5.4: Odd Parity

As you can seen, I tried to write my name in both cases, in odd parity and in the even parity, but some characters was appears whether the other was not, the explanation was for example if we take the character “S” which is equal “01010011” in binary you can see that the last bit is 1 so it’s an odd parity and that’s why its appear in the odd parity part, on the other hand if we take the character “R” which is equal “01010010” in binary you can see that the last bit is 0 so it’s an even parity and that’s why its appear in the even parity part.

4. Conclusion

This experiment was very important since we studied about Serial Communication with Arduino and its functions and how each of them works. We also learned how to connect two Arduino circuits and transmitting the data between them using UART circuit and how to control the push button and the LED using two Arduinos.

5. References

[1] https://learn.sparkfun.com/tutorials/what-is-an-arduino/all Access Date: 11-3-2021 at 11:00AM.

[2] https://www.circuitbasics.com/basics-uart-communication/ Access Date: 11-3-2021 at 11:10AM

[3] https://www.arduino.cc/reference/en/language/functions/communication/serial/ Access Date: 11-3-2021 at 1:30PM

[4] https://www.oreilly.com/library/view/arduino-cookbook/9781449399368/ch04.html Access Date: 11-3-2021 at 2:00PM

[5] Lab Manual.