LI-FI COMMUNICATION BETWEEN TWO ARDUINOS CARDS.

Li-Fi (Light Fidelity) is an advanced technology that allows data to be sent through optical communications such as visible light. Li-Fi data may travel by light and be interpreted at the receiver using any light-sensitive device, such as an LDR or a photodiode. Li-Fi connectivity may be 100 times faster than Wi-Fi.

In this project, we will demonstrate Li-Fi communication using two Arduino boards. On the transmitter side the data is sent with the help of a LED and a 4x4 keyboard. On the receiver side, we have a photoresistor (LDR) that captures this light coming from the LED.

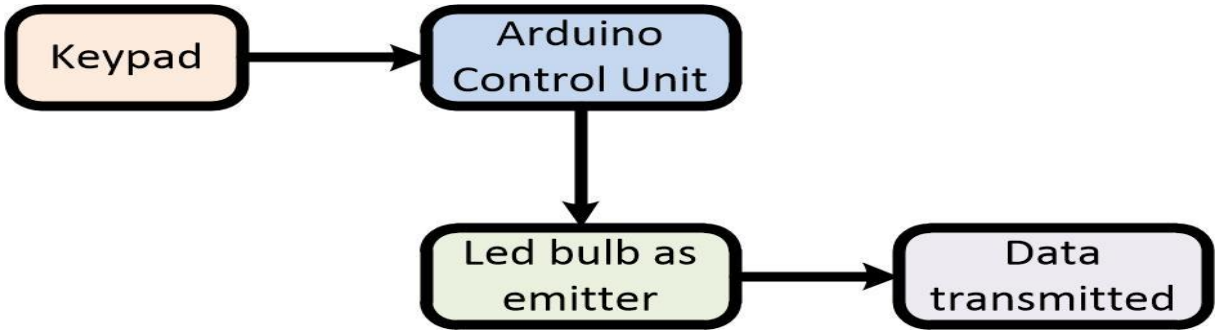
1. Components Required

- Arduino UNO
- Arduino MEGA
- LDR Sensor
- 4*4 Keypad
- 16*2 Alphanumeric LCD
- I2C Interface module for LCD
- Breadboard
- Connecting Jumpers
- LED

2. A brief overview of Li-Fi

For data transfer, Li-Fi makes use of visible light as a communication medium. A LED may function as a light source, and a photodiode can function as a photoreceptor, receiving and transmitting light signals. We may send distinct data models by controlling the luminous impulsion on the sensor's side. This phenomenon occurs at a very fast rate and cannot be seen by the human eye. Then, on the receiving end, the photodiode or photoresistance (LDR) converts the data into useful information.

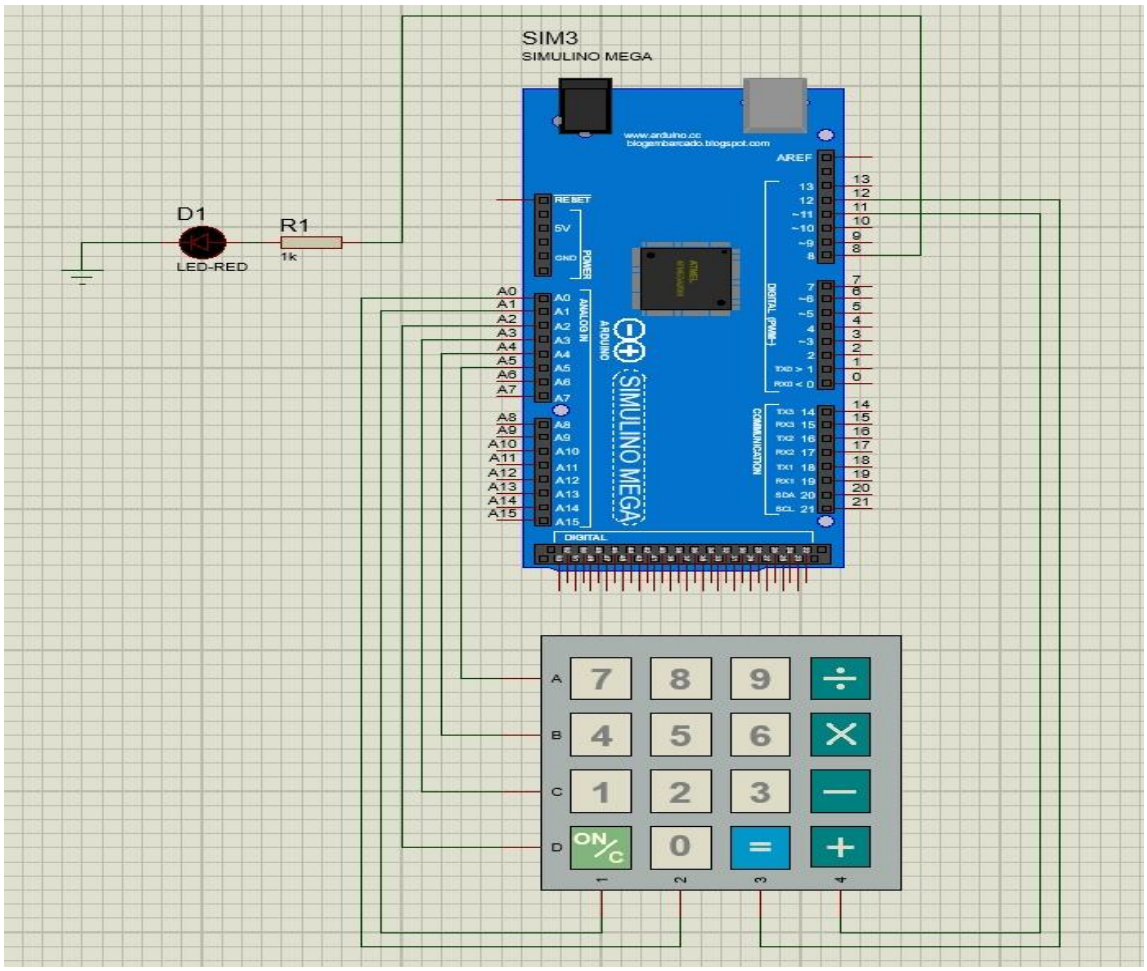
3. Li-Fi transmitter



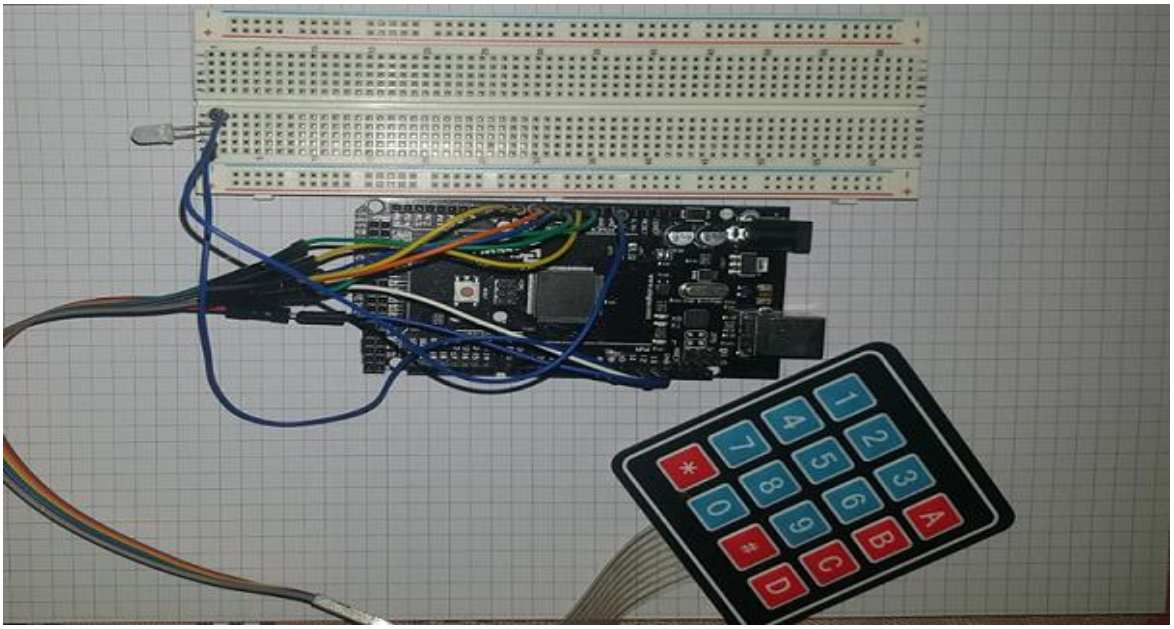
As seen in the diagram above, the emetteur in a Li-Fi communication is made up of a clavier that is used to collect information. This means we will use this keyboard to choose the data to be transmitted. The information is then processed by the control unit, which is nothing more than an Arduino card. This latter converts the data into binary impulsions, which are then

stored at the level of LED ampoules. These data are then sent via the visible light impulsions on the receiver's side.

4. Transmitter Side Circuit Diagram



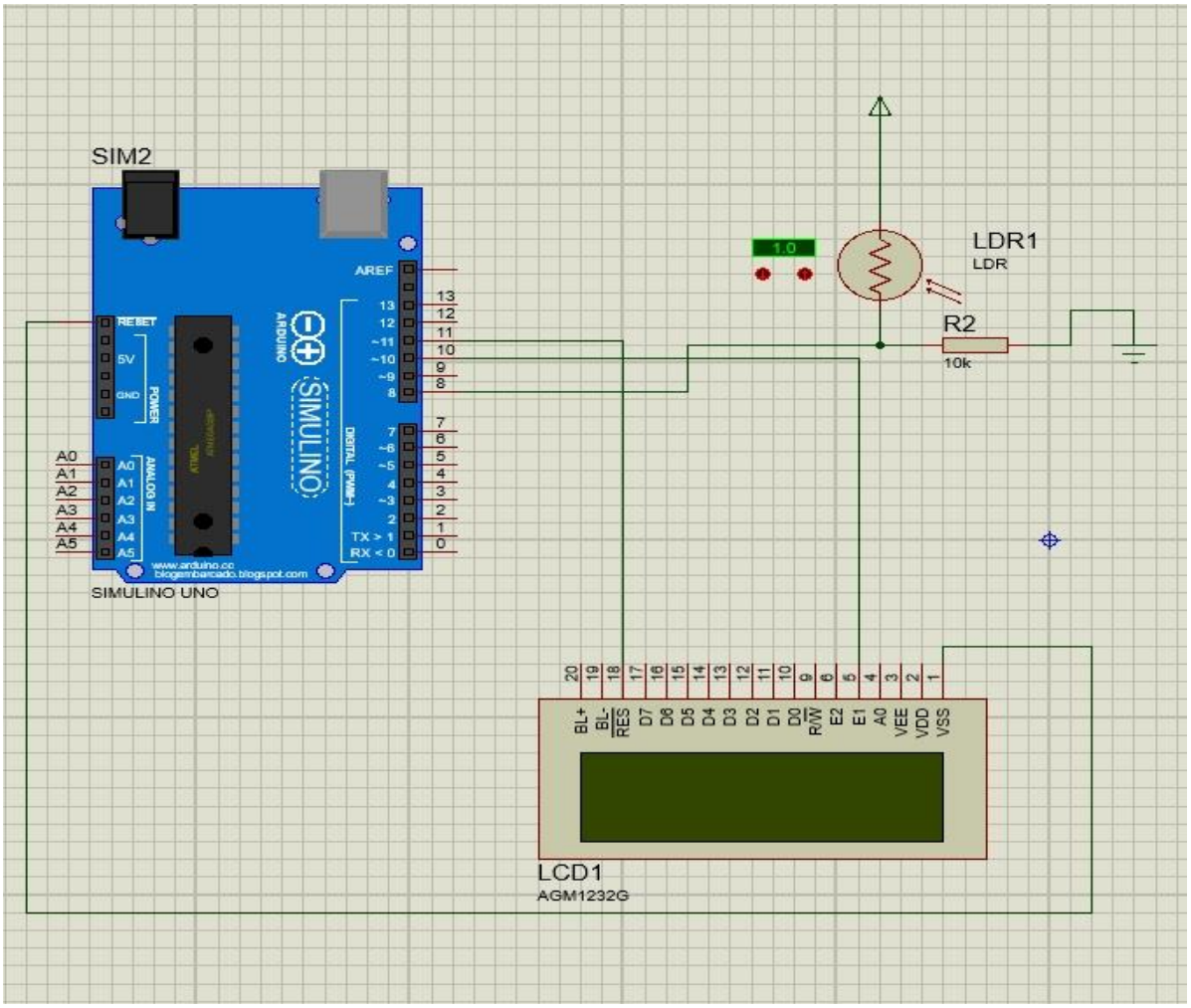
5. Physical Transmitter Installation



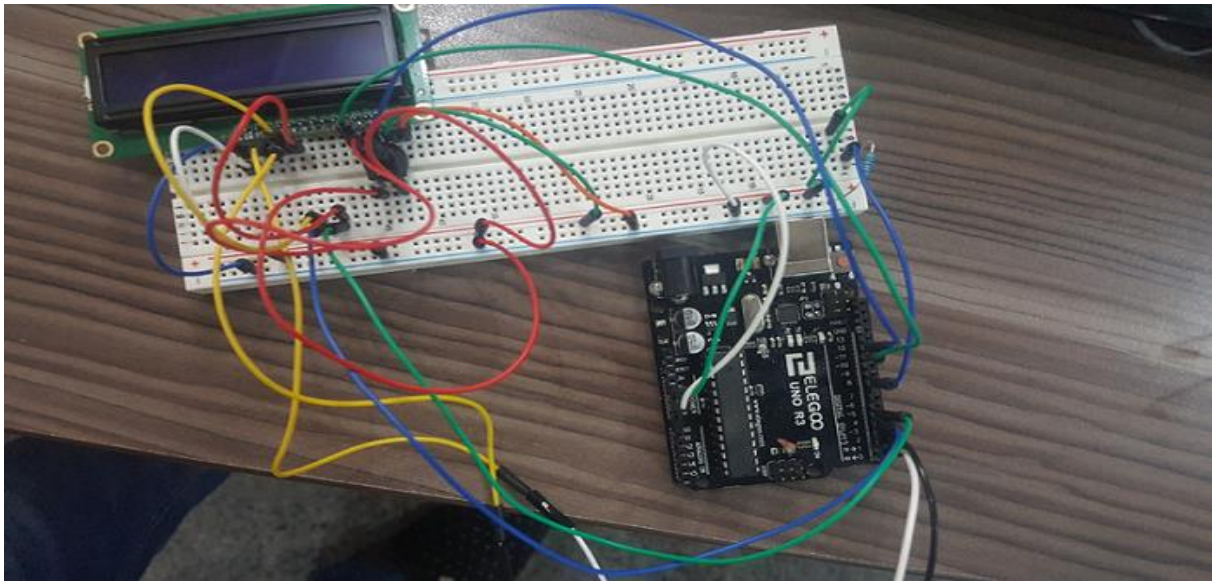
6. Li-Fi receiver

The LDR sensor in the receiver part receives the visible light pulses from the transmitter and converts them into readable electrical pulses, which are then transmitted to the Arduino (control unit). These pulses are received by the Arduino, which converts them into real-time data and displays them on a 16x2 LCD screen.

7. Receiver Side Circuit Diagram



8. Receiver Physical Installation



9. Implementation of the Li-Fi prototype

9.1. Arduino MEGA Li-Fi Transmitter Code :

As previously said, we have two Li-Fi components to implement : the transmitter and the receiver. The complete codes for each section are available via the link **https://github.com/Dib58/MyProject/tree/main**/, and an explanation of the codes is provided below :

The Arduino UNO is utilized on the transmitter side, along with a 4x4 keypad and an LED. To begin, all of the required library files are downloaded and installed on Arduino using the Arduino IDE.

```
#include <Keypad.h>
```

After successfully installing the library files, define the number of rows and column values, which is 4 for each because we are using a 4*4 keypad here.

```
const byte ROW = 4;
const byte COL = 4;
char keyscode[ROW][COL] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
```

The Arduino pins that are utilized to interface with the 4*4 keypad are then specified. In our situation, we utilized A5, A4, A3, and A2 for R1, R2, R3, and R4, and A1, A0, 12, 11, and C1 for C1, C2, C3, and C4.

```
byte rowPin[ROW] = {A5, A4, A3, A2};
byte colPin[COL] = {A1, A0, 12, 11};
Keypad customKeypad = Keypad( makeKeymap(keyscode), rowPin, colPin, ROW, COL);
```

Inside setup (), the output pin is defined, where the LED source is connected. Also, it is kept OFF while switching ON the device.

```
void setup()
{
 pinMode(8,OUTPUT);
 digitalWrite(8,LOW);
}
```

The values from the keypad are read using customKeypad inside the while loop. getKey() returns a value that is compared in the if-else loop to create distinct pulses for each key press. The code shows that the timer intervals are kept unique for all key values.

```
char customKey = customKeypad.getKey();
 if (customKey) {
  if (customKey == '1')
 {
  digitalWrite(8,HIGH);
  delay(10);
  digitalWrite(8,LOW);
 }
```

9.2. Arduino UNO Li-Fi Receiver Code :

The Arduino UNO is interfaced with an LDR sensor on the Li-Fi receiver side, as shown in the circuit diagram. In this case, the LDR sensor is connected in series with a resistor to produce a voltage divider circuit, and the analog voltage output of the sensor is sent to Arduino as an input signal.
We start by adding all the essential library files into the code, such as Wire.h for the LCD. There will be no need to download these libraries as they will be pre-installed with Arduino.

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
```

Configure the I2C module for 16*2 Alphanumeric LCD using the LiquidCrystal I2C class. In this scenario, we must supply the address, row, and column numbers, which are 0x3f, 16, and 2 accordingly.

```
LiquidCrystal_I2C lcd(0x3f, 16, 2);
```

Declare the pulse input pin to receive the signal inside setup (). Then, on the LCD screen, print a message "LIGHT FIDELITY (Li-Fi)" which will be displayed during setup.

```
void setup()
{
 pinMode(8, INPUT);
 Serial.begin(9600);
```

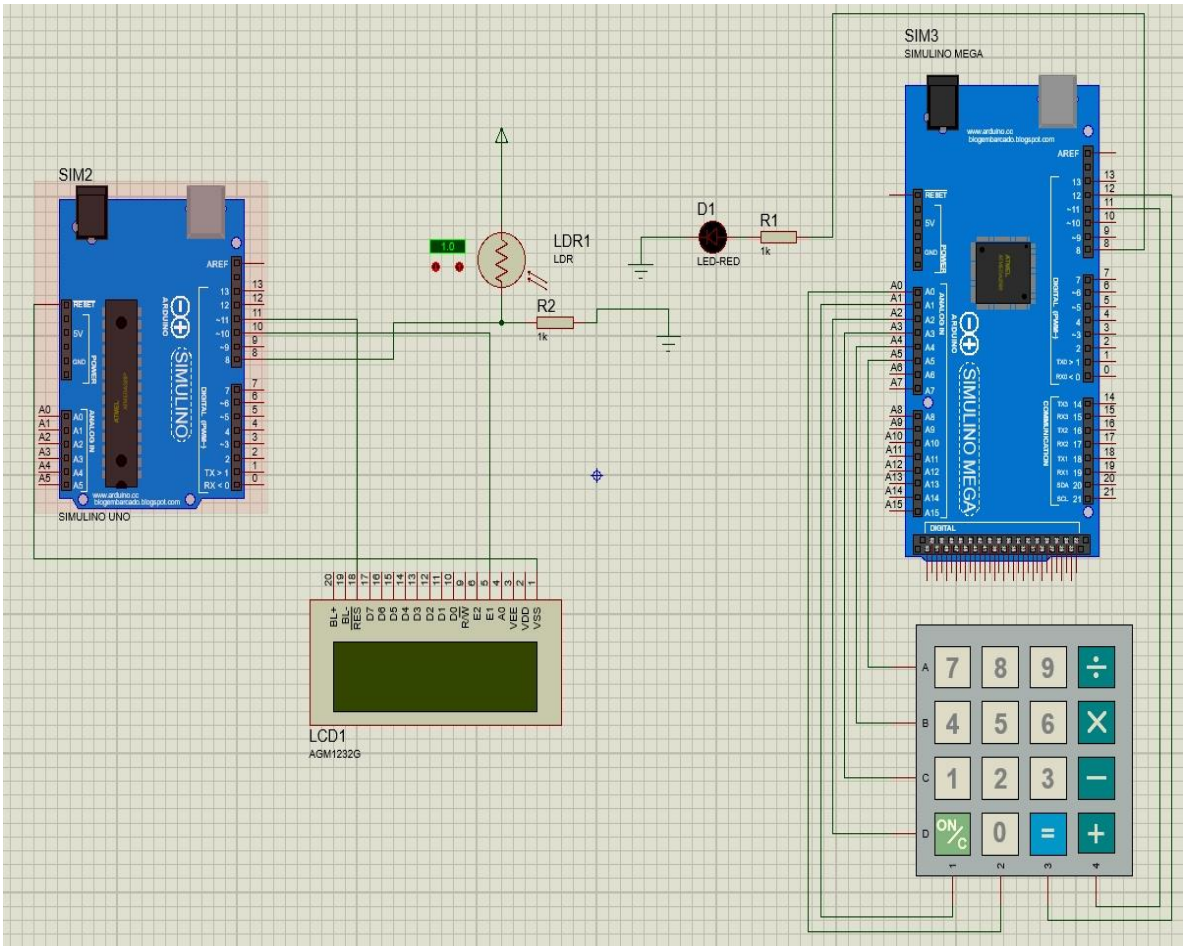
```
  lcd.init();
  lcd.backlight();
  lcd.setCursor(0, 0);
  lcd.print("  WELCOME TO ");
  lcd.setCursor(0, 1);
  lcd.print(" CIRCUIT DIGEST ");
  delay(2000);
  lcd.clear();
}
```

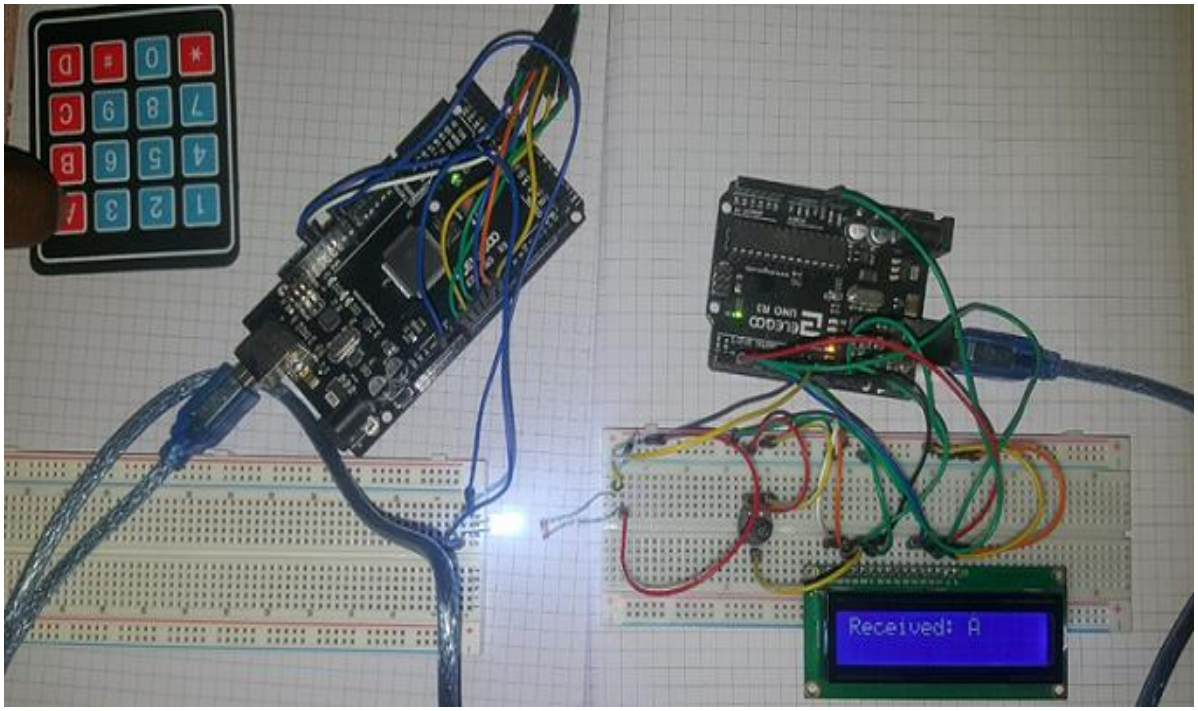
The pulse input time from the LDR is computed using the pulseIn function inside the while loop, and the kind of pulse is defined, which is LOW in our case. For debugging reasons, the value is reported on the serial monitor. It is recommended to double-check the duration, as it may vary depending on the arrangement.

```
unsigned long duration = pulseIn(8, HIGH);
 Serial.println(duration);
```

10. Li-Fi transmitter and receiver using Arduino

After uploading the entire code to both Arduinos, push any button on the keypad on the receiver side, and the identical digit will be shown on the 16x2 LCD on the receiver side.




Received: A