Sistemas Sensoriais
Arduino Report

Filip Szymański
Amadeusz Szymko
Bartosz Bednarski

1. Objective

The purpose of the class is to implement a solution to the following problem:

Using given hardware set up communication between „Slave” and „Master” computer and pass data continuously to the receiver (Master).

Hardware:
Arduino Uno Board
Bluetooth interface
Distance Sensor
Light Sensor

Our solution was implemented using Arduino provided software (Arduino v. 1.8.10).

2. Code

```
#include <SPI.h>
#include <RF24.h>


RF24 radio(7, 8); // CE, CSN
const uint64_t pipe = 0xE8E8F0F0E1LL;
char rdTxt[20] = "";
char respone[20] = "Test response";
const int trigPin = 5;
const int echoPin = 6;
float reply, cm, inch;
int sensorPin = A0; // select the input pin for LDR
int sensorValue = 0; // variable to store the value coming from the sensor

void radioRxMode() {
  radio.begin();
  radio.openReadingPipe(0, pipe);
  radio.setPALevel(RF24_PA_MIN);
  radio.startListening();
  pinMode(trigPin, OUTPUT); // Prepare the trigger pin
  pinMode(echoPin, INPUT); // Prepare the echo pin
}


void radioTxMode() {
  radio.begin();
  radio.openWritingPipe(pipe);
  radio.setPALevel(RF24_PA_MIN);
  radio.stopListening();
}


void setup() {
  // put your setup code here, to run once:
  Serial.begin(9600);
  radioRxMode();
}
```

```
void transfer(char* data){
    radioTxMode();
    Serial.println(rdTxt);
    Serial.println("Write");
    radio.write(&data, sizeof(data));
    memset(rdTxt, 0, sizeof(rdTxt));
}

void checkRadio(char* query){
  if(radio.available()){
    radio.read(&rdTxt, sizeof(rdTxt));
    if (strstr(rdTxt, query)){
      char transferData[20] =  "Test response";
      transfer(transferData);
    }else{
      Serial.println("Read");
      radioRxMode();
    }
  }
  delay(5000);
}


void time2Distance(long rawReply) {
    // The data sheet says that it takes sound 73.746 (almost 74) microseconds to travel 1 inch.
    // So if we divide rawReply which is in microseconds by the above we will get how many
    // inches the sound travelled forward and travelled back to the sensor.
    // The rawReply is the sound travelling to obstacle and back, then we need to divide by 2
    cm = rawReply/56.82;
}


void getDistance() {
    // Reset the trigger pin and get ready for a clean trigger pulse
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    // Generate and send to trigger pin the trigger pulse
    digitalWrite(trigPin, HIGH); // You need to keep it high for 10 micro seconds length
    delayMicroseconds(10); // This is the 10 microseconds we mentioned above :)
    digitalWrite(trigPin, LOW); // Stop the trigger pulse after the 10 microseconds
    // Now let's see how long did it take the sound wave to travel
    // It will take a time depending on the distance to the obstacle
    // This time may be up to 38 millisecond in case of no obstacle
    reply = pulseIn(echoPin, HIGH);
    // Now we have the reply in microseconds, but we need a distance!
    time2Distance(reply);
    // Let's print the distance to the Arduino's Serial Monitor. Tools --> Serial Monitor
    Serial.print("Distance in cm ");
    Serial.println(cm);
}


void getLight(){
    sensorValue = analogRead(sensorPin); // read the value from the sensor
    Serial.print("Value of LDR sensor: ");
    Serial.println(sensorValue); //prints the values coming from the sensor on the screen
}


void loop() {
    // put your main code here, to run repeatedly:
    //char query[20] = "Query";
    //checkRadio(query);
    //getDistance();
    getLight();
    delay(1000);

}
```

3. Measurements

In order to make sure whether the system works properly we had to measure some values and validate them.

To test the distance sensor we placed it on stable and even surface. Then we put an object in front of the sensor and measured the distance between the sensor and the object.

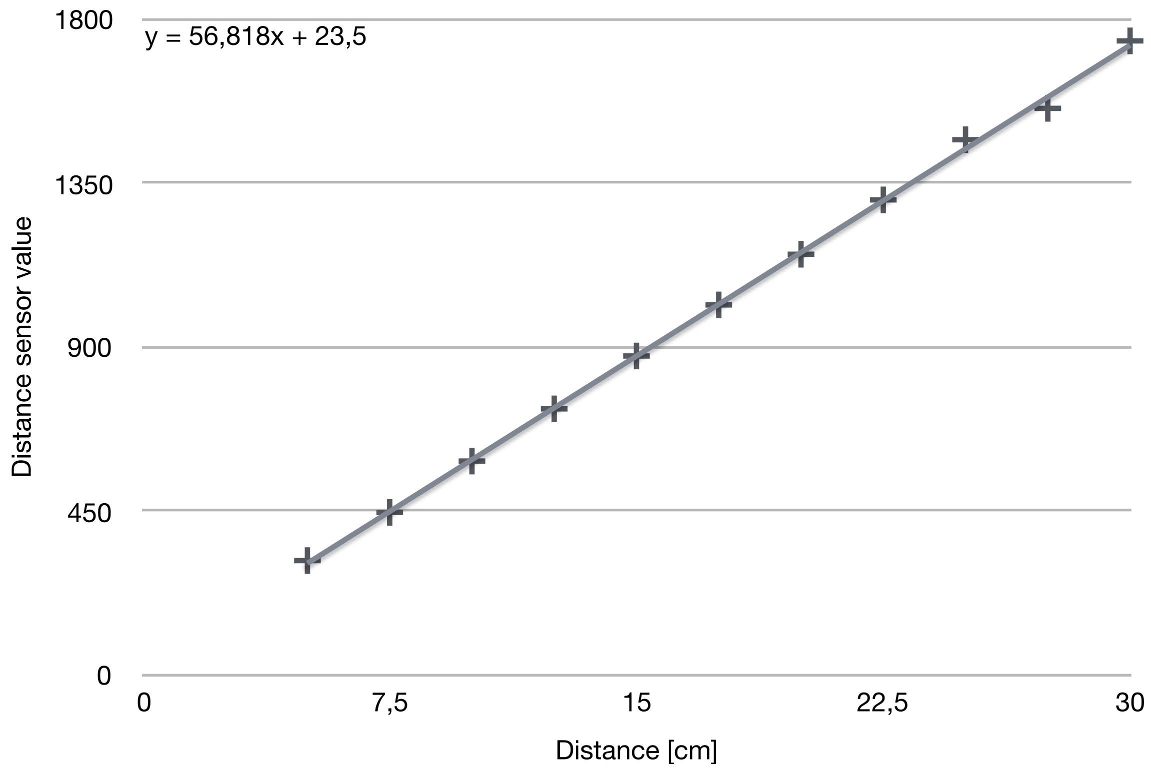
When it comes to the light sensor we used a table lamp and a photometer. W placed it next to our light sensor and gathered measured data from both devices for different lamp (lightbulb) positions (different distance from the sensors).

4. Results

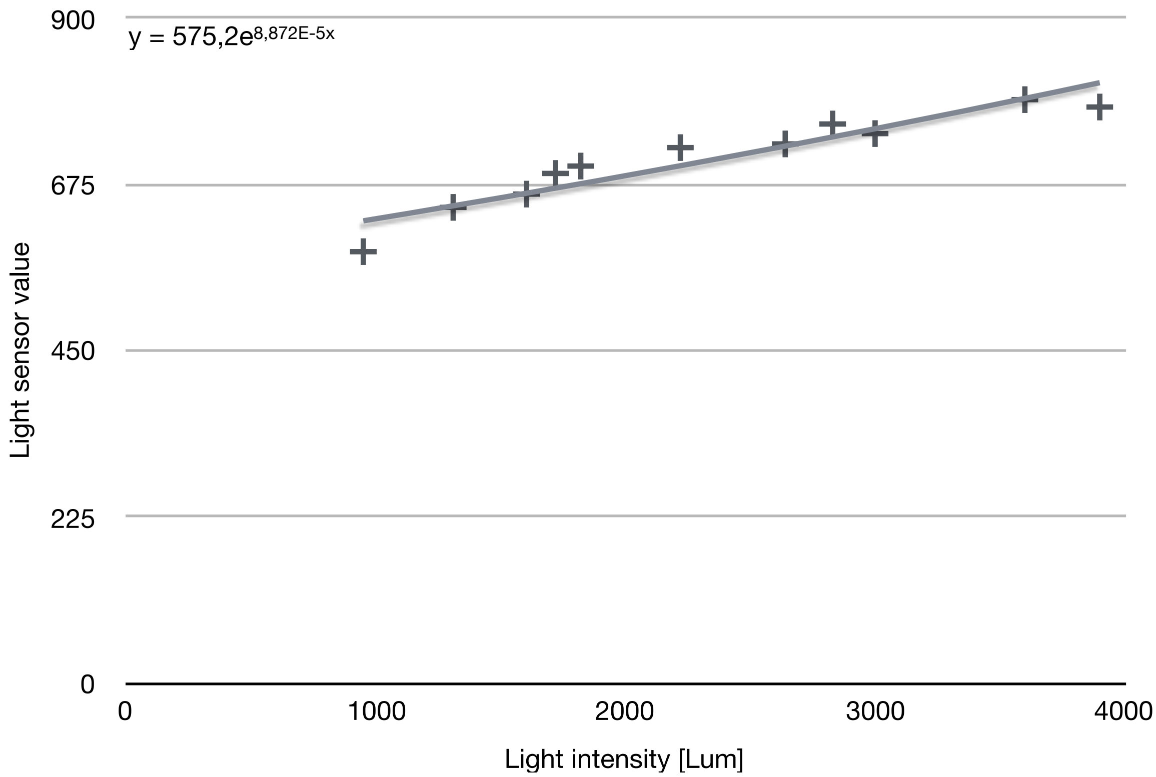
Distance [cm]	Sensor value
5	315
7.5	447
10	588
12.5	731
15	876
17.5	1016
20	1155
22.5	1304
25	1469
27.5	1555
30	1740

Light [lum]	Sensor value
950	584
1310	644
1604	662
1720	690
1821	700
2220	725
2640	730
2830	757
3000	744
3600	790
3900	780

4. Charts



1800
1350
900
450
0
y = 56,818x + 23,5
Distance sensor value
0
7,5
15
22,5
30
Distance [cm]



900
675
450
225
0
y = 575,2e^8,872E-5x
Light sensor value
0
1000
2000
3000
4000
Light intensity [Lum]

5. Summary

After data was gathered we calculated trend lines for both cases.
For the distance sensor we used linear regression. Points seem to fit the line pretty well and it looks like the line would cross the start of the coordinate system (point 0,0).

When it comes to the light measurement we used an exponential function. The line seems almost straight but it is probably because the differences between the highest and lowest measure is too small.
Also the measurements took place in a classroom, not in a completely dark room so there is a chance it interfered with gathered data.

Despite some issues with light measurement the experiment was successful as the purpose of the class was the setup of the Arduino hardware and implementation of the software.