Car sensors project

Author: Mihai Ghise

Contents

Chapter 1

Summary

Problem Statement and Goals: Building a system that consists of Arduino sensors resembling some basic car sensors like rain sensor, light sensor, and proximity sensor. This idea of the project is very useful to gain some experience in designing and building different stuff using Arduino. The project can be used in the car industry for the following areas: rain sensor, detects water on it, and power up the wipers, light sensor, detects the insufficient ambient light and starts the headlights of the car, and proximity sensor, detects and warn the driver if there are close objects in front of the car.

Here is a list of components that were used in order to build my project:

- Arduino Mega 2560 board
- Ambiental light sensor, cheap sensor, very useful when it comes to detecting the strength of the outside light - used to lit up some white LEDs when the threshold set for light is passed, the only downside of this sensor is that it's light detector part of it breaks down really fast if it isn't carefully used
- Water sensor, cheap as well, durable, used for the ”wipers” thing
- Proximity sensor, used for reading the distance between the system and an object in front of it, I had problems with this sensor because, sometimes it just went crazy if there was just a little bit of dirt on it
- Buzzer, connected to the proximity sensor, starts beeping when the distance between the system and an object is smaller than a threshold
- Leds of different colors

Chapter 2

Schematic

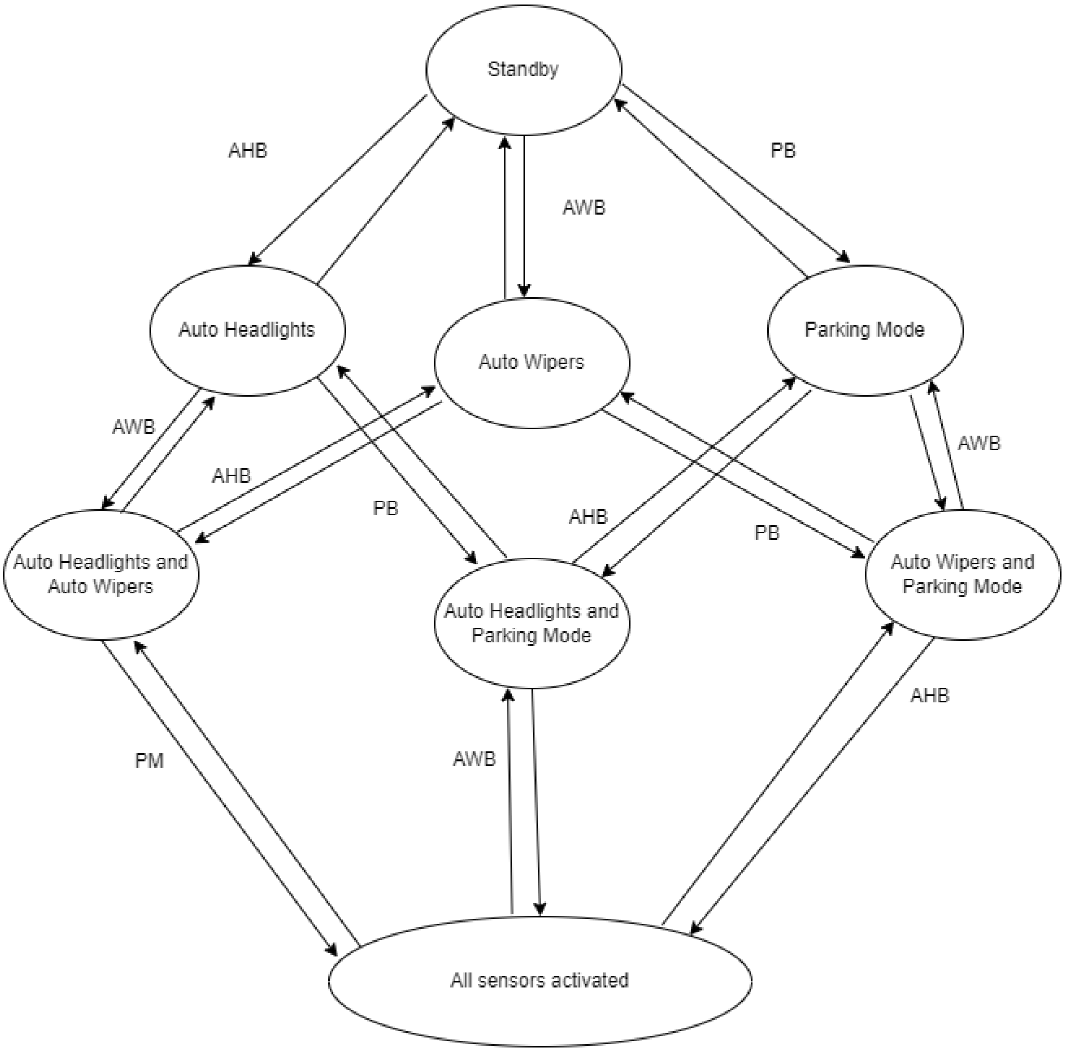

Figure 2.1: State diagram

Nomenclature:

- AHB = Auto Headlights Button
- AWB = Auto Wipers Button
- PB = Parking Button

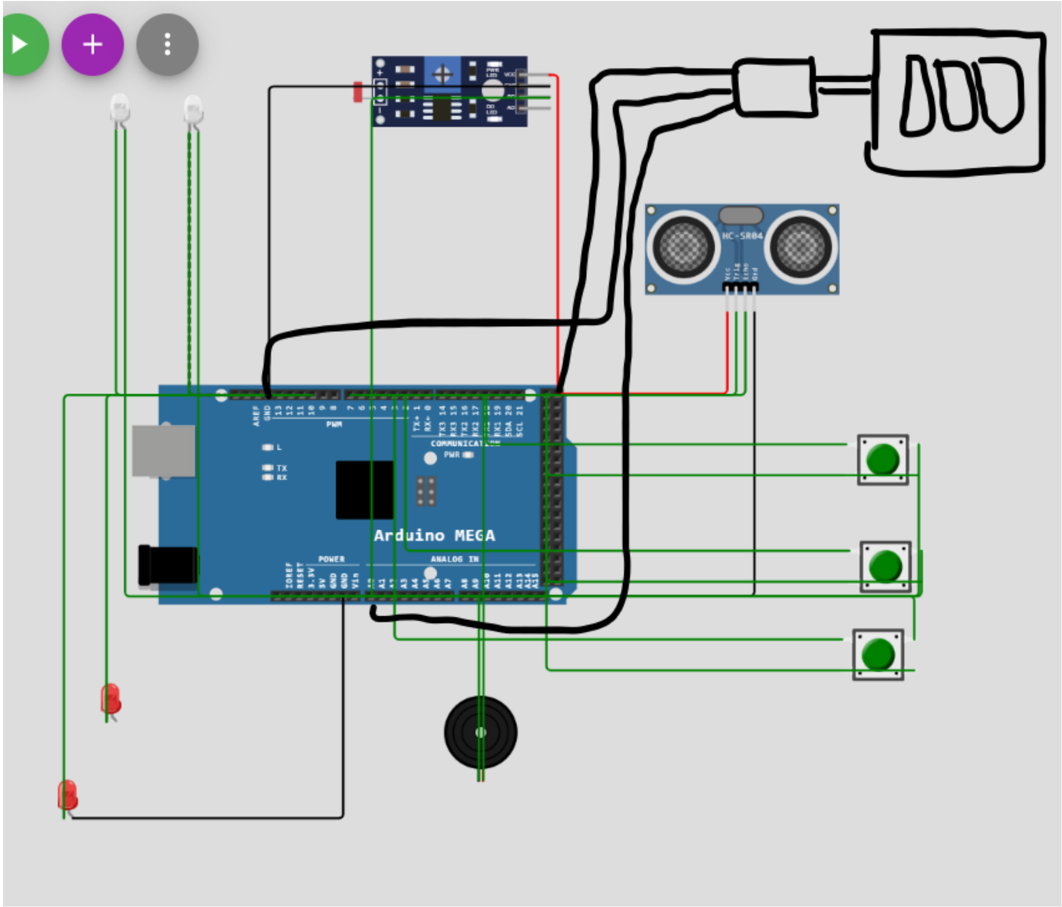

Figure 2.2: Circuit Schema

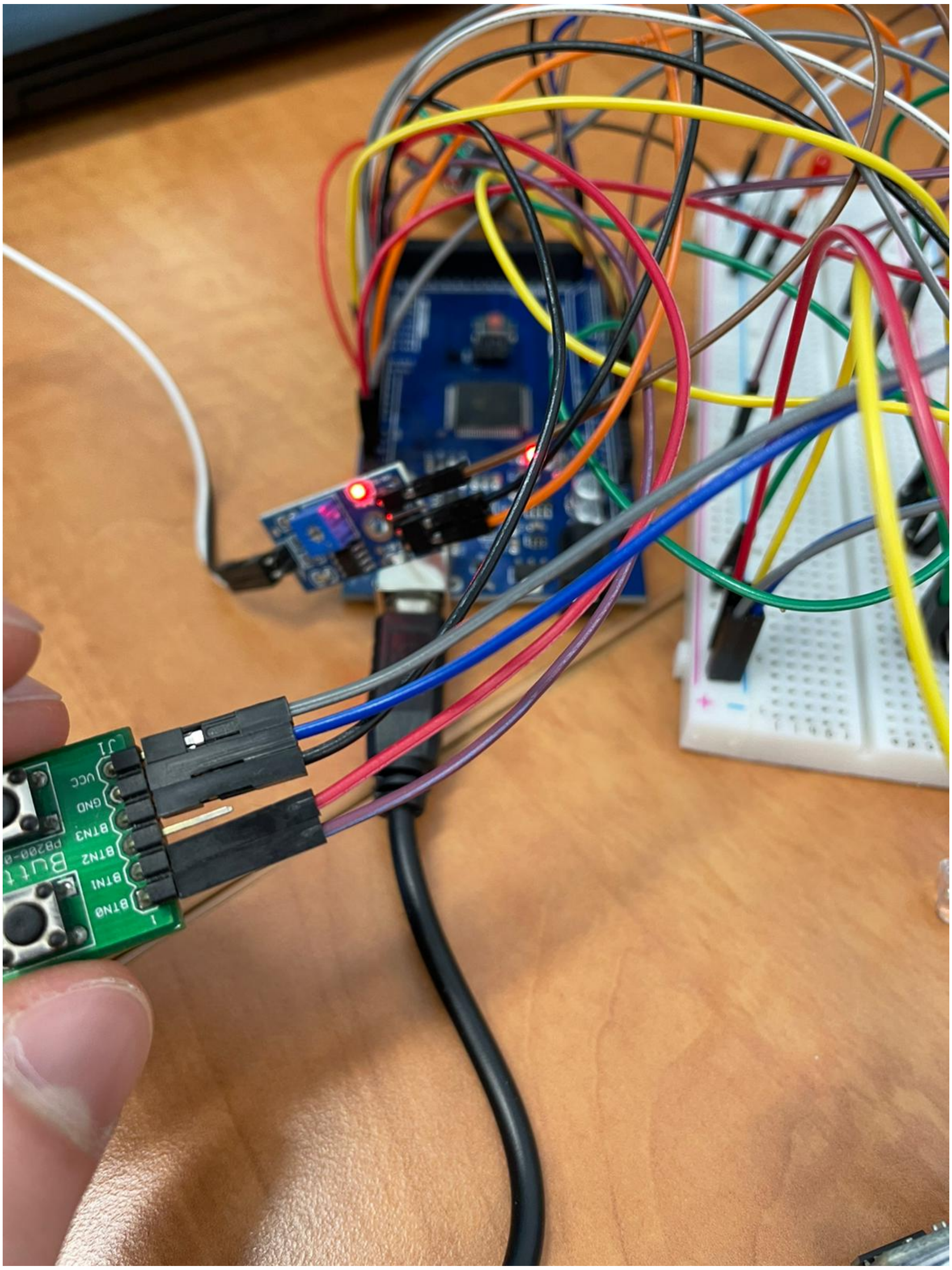

Figure 2.3: Picture 1 of the system

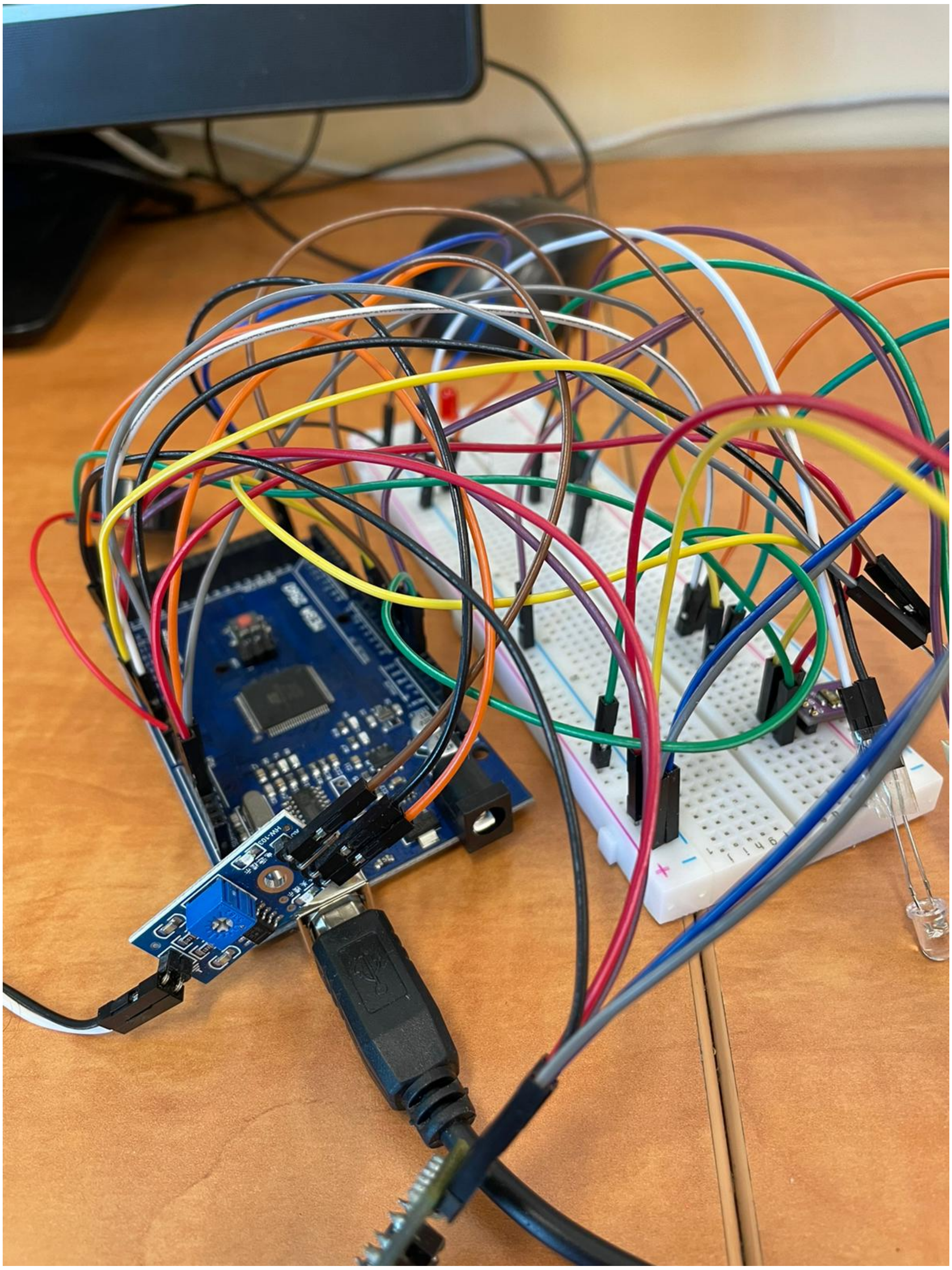

Figure 2.4: Picture 2 of the system

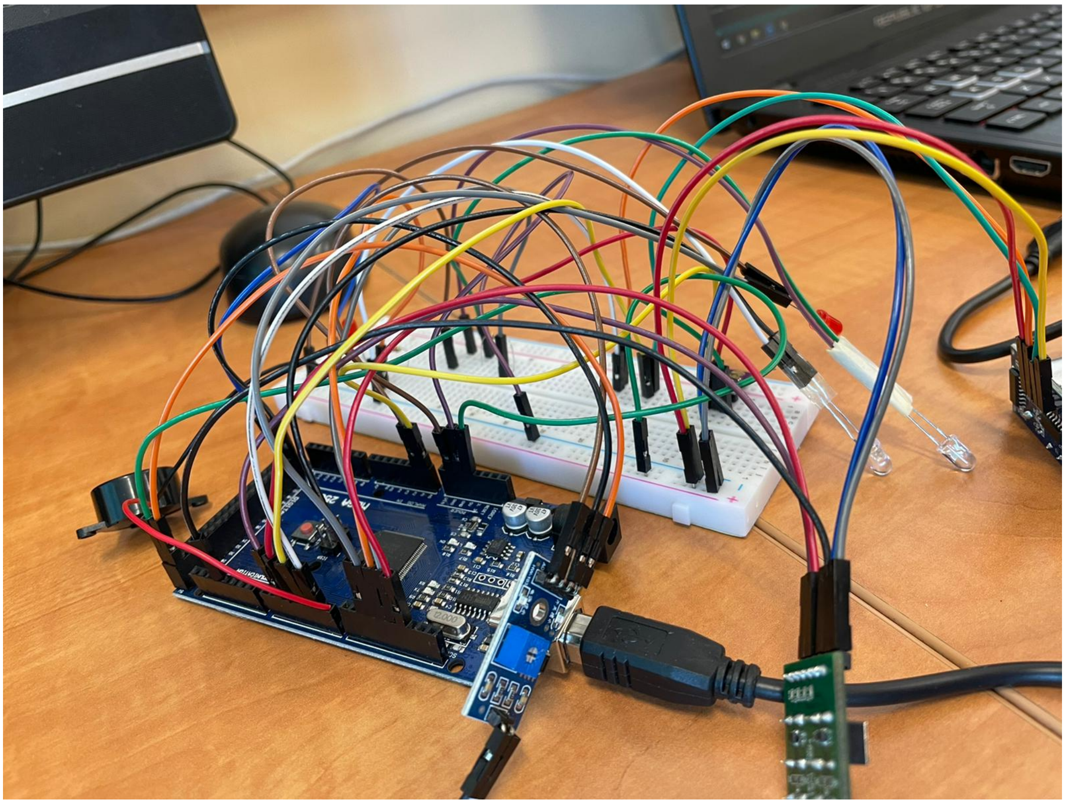

Figure 2.5: Picture 3 of the system

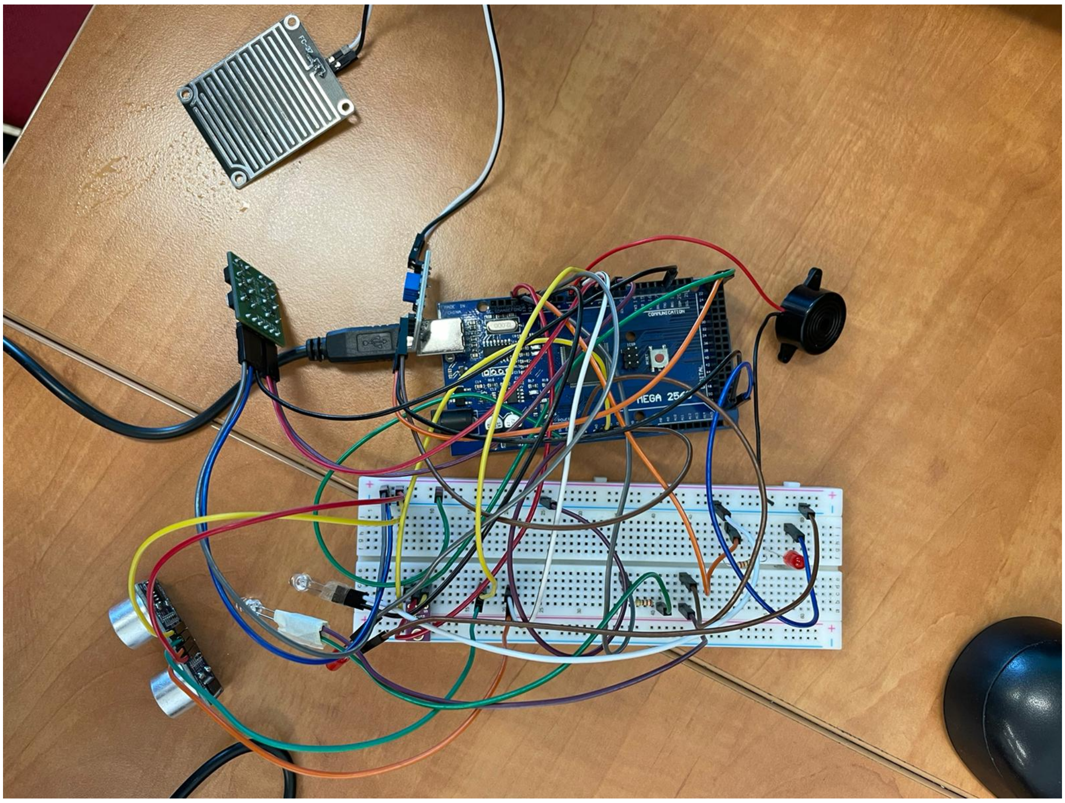

Figure 2.6: Picture 4 of the system

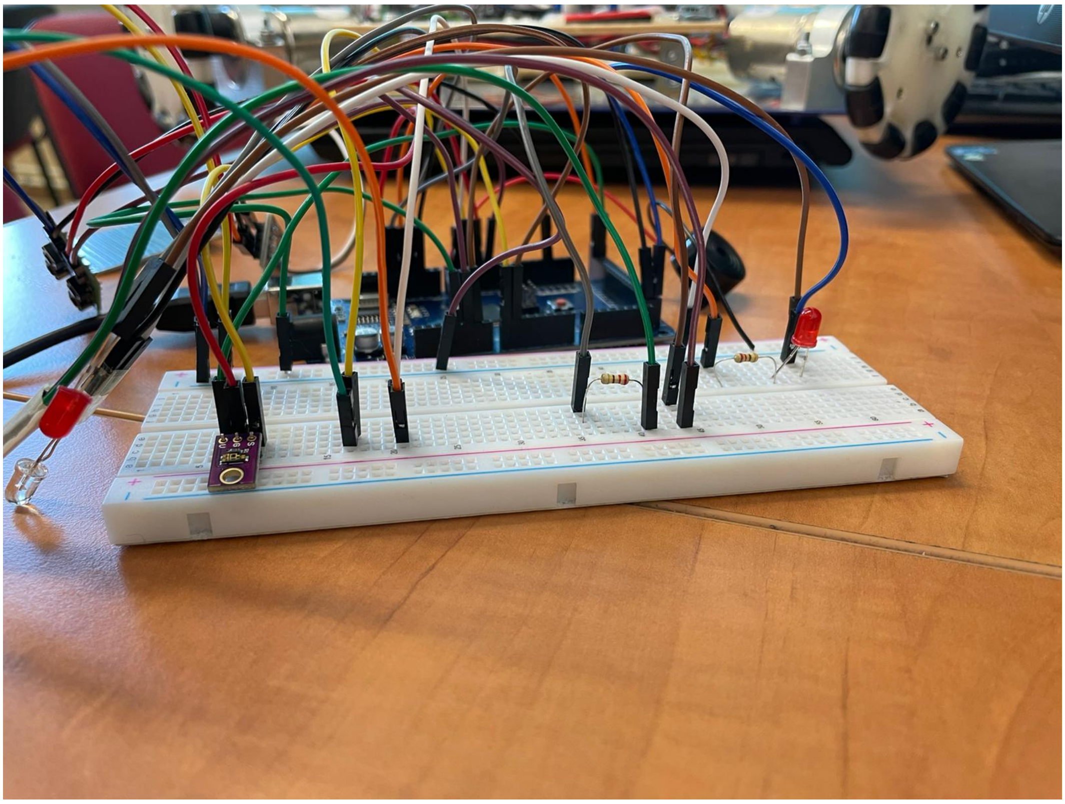

Figure 2.7: Picture 5 of the system

Chapter 3

Project explanation

So basically the system runs like a car. It has three buttons, one for the headlights, one for the wipers, and one for parking assistance. Normally, the buttons activate some interrupts when they are pressed. When the system starts, nothing happens. Imagine this, in your car, you have an auto wiper button, that's the wipers button, if you press that in my system, the water sensor starts reading information and if the data read is under a threshold, that means there is water on the sensor, there is a red LED that is lit up(symbolizing that the wipers are working, since I can not use big 2 meters wipers in my project).

Then, for the light sensor, just when its button is pressed, it will start reading information. Remember that in most cars, at night, if you do not switch to AUTO HEADLIGHTS, your car will not start the headlights. Same here, the button on my system, represents that switch in most of the cars. So, it works like this, you press the button, the sensor starts working and if the value read is not big enough, which means there isn't enough light outside, two white LEDs are lit up which represent the headlights of the car.

The proximity sensor works the same, it has a button, an interrupt attached to it, switching from reading data to not reading data. The sensor detects and calculates the distance between it and an object placed in front of it and then, if the distance is smaller than 5 cm, a buzzer will start beeping so that the driver is warned that he is closer to an object and might scratch his car.

Chapter 4

References

Listing 4.1: Arduino Code

```
volatile boolean headlights = false; //bool var for headlights button
volatile boolean wipers = false; //bool var for wipers button
volatile boolean parking = false; //bool var for parking sensor button


const int buzzer = 7; //pin for the buzzer
const int ledPin = 8;
const int trigPin = 6;
const int echoPin = 5;


const int autoHeadlightsPin = 12;
const int ledHeadlightsPin = 13;
const int ledHeadlightsPinSecond = 10;
const int buttonHeadlightsPin = 2;

const int autoWipersPin = 11;
const int ledWipersPin = 14;
const int buttonWipersPin = 3;

const int buttonParkingPin = 18;


long duration;
int distance, rain, light, safetyDistance;

void setup()
{
  Serial.begin(9600);
  pinMode(ledHeadlightsPin, OUTPUT); //led for headlight
  pinMode(ledHeadlightsPinSecond, OUTPUT); //led for second headlight
  pinMode(buttonHeadlightsPin, INPUT); //button for headlights
  attachInterrupt(digitalPinToInterrupt(buttonHeadlightsPin), headlightsButton, RISING); //interrupt for headlights button

  pinMode(ledWipersPin, OUTPUT); //led for wipers
  pinMode(buttonWipersPin, INPUT); // button for wipers
  attachInterrupt(digitalPinToInterrupt(buttonWipersPin), wipersButton, RISING); //interrupt for wipers button

  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  pinMode(buzzer, OUTPUT);
  pinMode(ledPin, OUTPUT); //led for parking sensor
  pinMode(buttonParkingPin, INPUT); //button for parking sensor
  attachInterrupt(digitalPinToInterrupt(buttonParkingPin), parkingSensor, RISING); // interrupt for parking sensor
}

void loop()
{
  light = analogRead(A0);
  rain = analogRead(A1);

  //check if auto headlights is on
  if (headlights == true)
  {
    Serial.println(light);
    //check if there is enough light
    if (light < 90)
    {
      digitalWrite(ledHeadlightsPin, HIGH);
      digitalWrite(ledHeadlightsPinSecond, HIGH);
      Serial.println(”Headlights activated.”);
    }
    else
    {
      digitalWrite(ledHeadlightsPin, LOW);
      digitalWrite(ledHeadlightsPinSecond, LOW);
    }
  }


  //check if auto wipers is on
  if (wipers == true)
  {
    Serial.println(rain);
    //check if the is water on the sensor
    if (rain > 400)
    {
      digitalWrite(ledWipersPin, LOW);
    }
    else
    {
      digitalWrite(ledWipersPin, HIGH);
      Serial.println(”Wipers activated.”);
    }
  }

  if (parking == true)
  {
    distance = calculateDistance();

    Serial.print(”Distance : ”);
    Serial.println(distance);

    safetyDistance = distance;

    if (safetyDistance <= 5)
    {
      digitalWrite(ledPin, HIGH);
      Serial.println(”BUZZZZZ”);
      tone(buzzer, 450);
      delay(100);
      noTone(buzzer);
      delay(100);
      safetyDistance = calculateDistance();
    }
  }
  delay(300);
}

int calculateDistance()
{
  //make sure the trig pin is empty
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

  //generate ultrasound wave
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = duration * 0.034 / 2;

  return distance;
}

//interrupt function for headlights button
void headlightsButton()
{
  if (headlights == true)
  {
    headlights = false;
    digitalWrite(autoHeadlightsPin, LOW);
    digitalWrite(ledHeadlightsPin, LOW);
    digitalWrite(ledHeadlightsPinSecond, LOW);
  }
  else
  {
    headlights = true;
    digitalWrite(autoHeadlightsPin, HIGH);
  }
}

//interrupt function for wipers button
void wipersButton()
{
  if (wipers == true)
  {
    wipers = false;
    digitalWrite(autoWipersPin, LOW);
  }
  else
  {
    wipers = true;
    digitalWrite(autoWipersPin, HIGH);
  }
}

//interrupt function for parking sensor
void parkingSensor()
```

```
{
  if (parking == true)
  {
    parking = false;
    //digitalWrite(ledPin, LOW);
  }
  else
  {
    parking = true;
    //digitalWrite(ledPin, HIGH);
  }
}
```