EMS PROJECT

RED LIGHT

GREEN LIGHT

ZOTESCU RUTH
TATARAU OANA
BORODI BIANCA-IOANA
OROSZ BARBARA

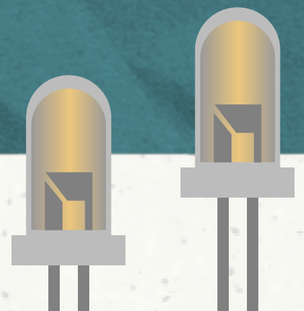
TODAY'S AGENDA

1. About the Game
2. The Rules
3. Technical Details
4. The Circuit
5. The Code
6. Thank You

ABOUT THE PROJECT

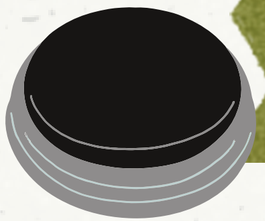
This project introduces an interactive motion-based game designed after the classical game “Red Light Green Light”. Employing: two distance sensors, two LEDs, a buzzer and a button.

Two LEDs as visual indicators, providing immediate feedback to players when they lose.



Two distance sensors with a rapid sampling rate for real-time motion detection, sensors are strategically placed to cover the game area effectively, one for each player.

A buzzer to signal the start, pause, and end of each game round.



A button for the players to push when are near the bord for them to mark their win.

THE RULES OF THE GAMES

Players take positions at the starting point within the game area.
The buzzer initiates the game, signaling the start of motion for players.

Players must immediately cease movement. Any detected motion for the distance sensors in the freeze phase leads to player elimination from the ongoing round.

At specific intervals, the buzzer stops, marking the freeze phase, the buzzer will buzz faster when the time comes so the players will have a signal before the change.

The buzzer will make a short sound to signal the end and the LED near the lousing player will turn on to signal how moved. To win the game a player has to go near the bord and push the button.

TECHNICAL DETAILS

COMPONENTS:

- NodeMCU ESP8266MOD as the Unit's Controller
- Analog and Digital Distance Sensors
- Buzzer
- Two LED-s
- Button
- Resistances
- Wires

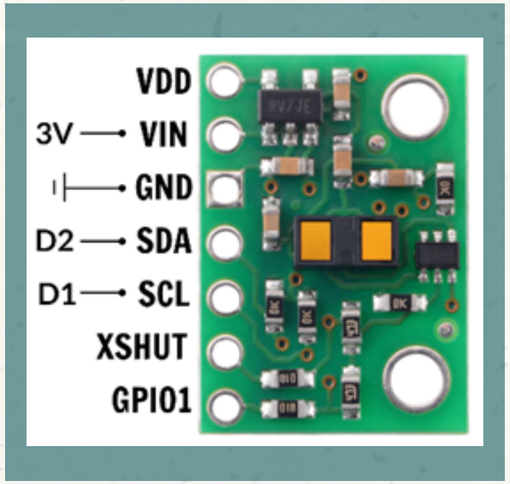
SENSORS

VL53L1X DIGITAL DISTANCE SENSOR

-works on the time-of-Flight principle (measures the time an emitted infrared signal takes from the sensor to the target and back)

-Laser-ranging sensor, offers fast and accurate ranging up to 4 m.

-SPAD (single photon avalanche diode) – photodetector designed to detect extremely low levels of light, down to the level of single protons



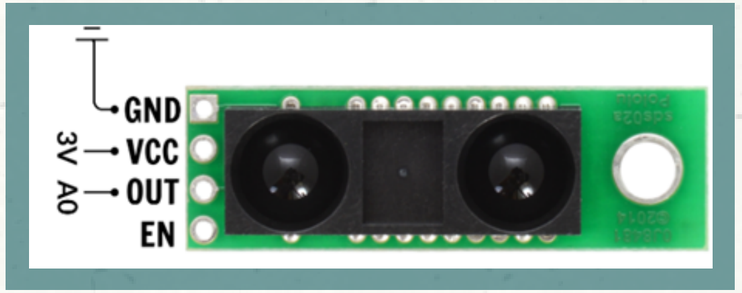
SHARP GP2Y0A60SZLF ANALOG DISTANCE SENSOR

-works on the time-of-Flight principle

-analog output converted into digital signal by the board

-distance measuring range: 10 – 150 cm

-sampling rate: 60Hz



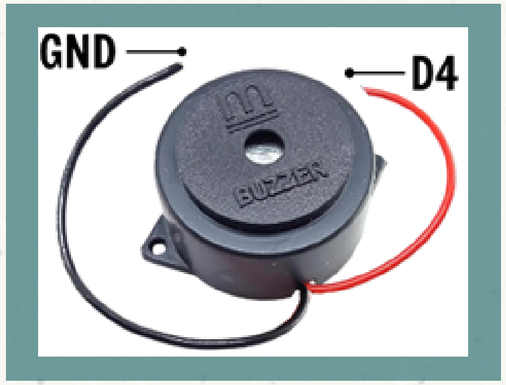
BUTTON

The game begins only when the button is pressed, and the initial beeping stops. It can be interrupted by pressing the button again, returning the unit to its initial state.

ACTUATORS

LED-S

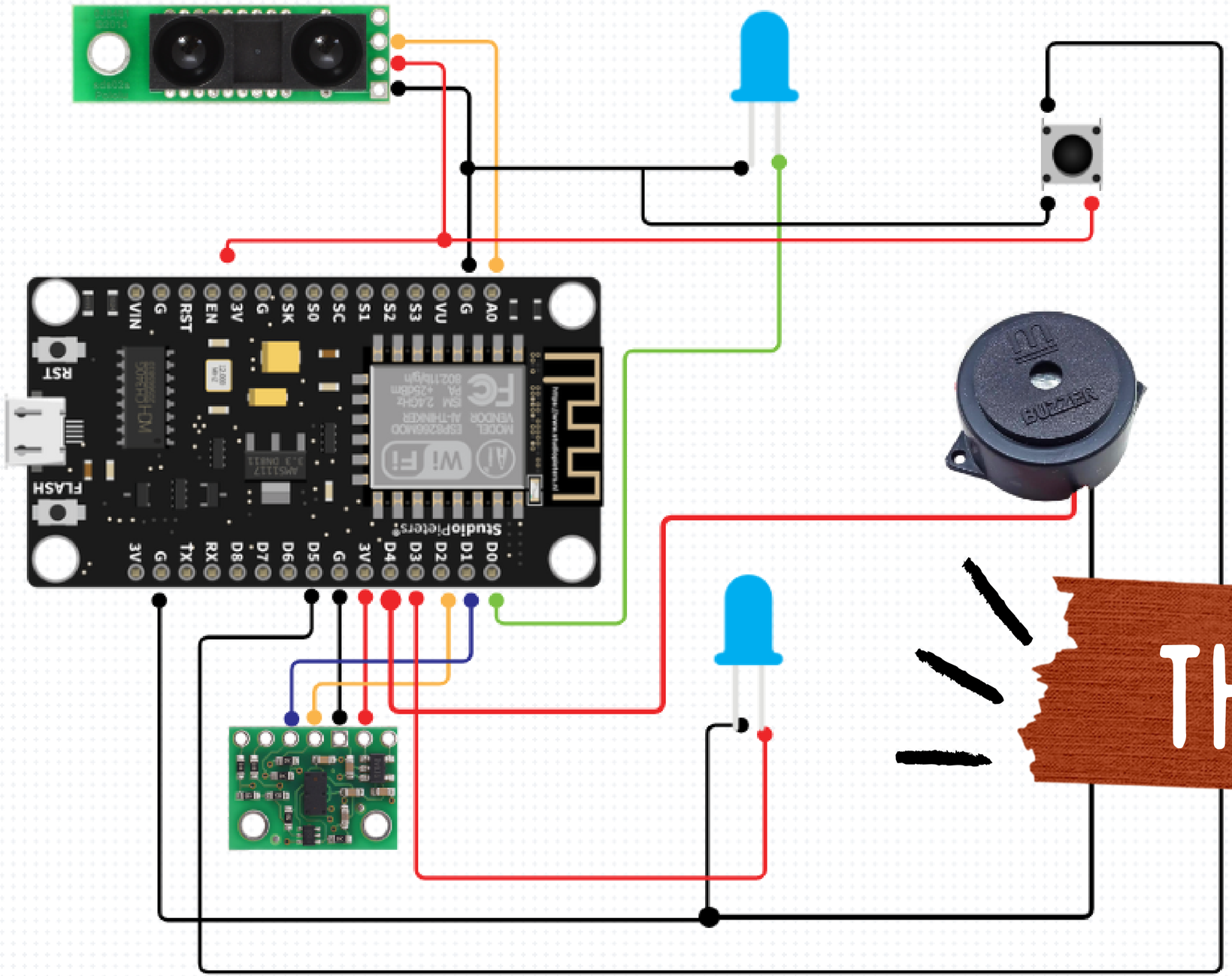
Each LED is paired to a sensor that blinks when triggered by movement when the buzzer is silent.
Each LED has a series connection with a resistance of 470 +- 23.5 Ω.



BUZZER

A piezoelectric buzzer produces sound through the piezoelectric effect, which generates an electric charge in response to mechanical stress.
This stress is caused by an oscillating crystal or ceramic element.

The buzzer's role is to signal the player the safe moment in which it can move towards the sensor. Its silence alerts that the sensors are on and are detecting the movement.


VIN
G
RST
EN
3V
G
SK
SO
SC
S1
S2
S3
VU
G
A0
RST
FLASH
Wi Fi
StudioPieters®
3V
G
TX
RX
D8
D7
D6
D5
G
3V
D4
D3
D2
D1
D0
BUZZER

THE CIRCUIT

THE CODE

1. INITIALIZATION (SETUP FUNCTION):

Purpose: Set up the initial configuration of the pins and sensors.

-Initialize pins for LEDs, button, and buzzer.

-Begin Wire communication for I2C sensors.

-Initialize the VL53L1X sensor, checking for successful initialization.

-Print a message indicating the status of the sensor.

```
#include <MedianFilter.h>
#include <SharpDistSensor.h>

#include <Wire.h>
#include "SparkFun_VL53L1X.h"

#define SHUTDOWN_PIN 2
#define INTERRUPT_PIN 3

// Pin configuration
const int sensorPin = A0; // Analog input pin

SFEVL53L1X distanceSensor;
SharpDistSensor sensor(sensorPin);

int ledPinA = D0;
int ledPinD = D3;
int buttonPin = D5;
int analogValue1 = 0;
int analogValue2 = 0;
int digitalValue1 = 0;
int digitalValue2 = 0;
int buttonState = 0;
int start = 0;
int buzzerPin = D4;
```

```
void setup() {
  pinMode(buttonPin,INPUT);
  pinMode(ledPinA, OUTPUT);
  pinMode(ledPinD, OUTPUT);
  pinMode(buzzerPin, OUTPUT);
  Wire.begin();

  Serial.begin(115200);
  //Serial.println("VL53L1X Qwiic Test");

  if (distanceSensor.begin() != 0) //Begin returns 0 on a good init
  {
    Serial.println("Sensor failed to begin. Please check wiring. Freezing...");
    while (1)
      ;
  }
  Serial.println("Sensor online!");
}
```

2.GAME START (LOOP FUNCTION):

Purpose: Check for the button press to start the game.

- Use a delay and buzzer to create a countdown effect for the start of the game.
- Set a flag (start) to indicate the game has 'started.'indicating the status of the sensor.

```
void loop(void) {
  delay(500);
  for (int k=0;k<10;k++)
  {
    delay(100);
    digitalWrite(buzzerPin, HIGH);
    delay(100);
    digitalWrite(buzzerPin, LOW);

  }

  buttonState = digitalRead(buttonPin);
  if( buttonState==HIGH)
    start=1;
  while( start==1)
  {
    digitalWrite(buzzerPin, HIGH);
    delay(1000);
    digitalWrite(buzzerPin, LOW);
    for (int k=0;k<3;k++)
    {
      delay(100);
      digitalWrite(buzzerPin, HIGH);
      delay(100);
      digitalWrite(buzzerPin, LOW);
    }
```

3.SENSOR READINGS AND GAME LOGIC (LOOP FUNCTION):

Purpose: Obtain readings from both analog and digital sensors and implement game logic.

-Use a loop to average readings from the analog sensor over a short period.

-Start ranging on the VL53L1X sensor and wait for data readiness.
-Get distance readings from the digital sensor.
-Compare analog and digital readings to detect player movement.
-Activate LEDs and buzzer based on the movement detected.
-Display messages and control indicators based on the sensor readings.

```
void loop(void) {
for(int i = 0; i < 10; i++)
   {
     analogValue1=0;
     digitalValue1=0;
     analogValue2=0;
     digitalValue2=0;
     Serial.print("\n-----------------------------------\n ");
     for (int j=0;j<10;j++)
     {
       analogValue1 += sensor.getDist();

       delay(10);
     }
     analogValue1/=10;
     Serial.print("A1: ");
     Serial.print(analogValue1);
      distanceSensor.startRanging();
     while (!distanceSensor.checkForDataReady())
     {
       delay(1);
     }
      digitalValue1 = distanceSensor.getDistance();
     distanceSensor.clearInterrupt();
     distanceSensor.stopRanging();

     Serial.print("\t\tD1: ");
     Serial.print(digitalValue1);
```

```
  Serial.println();
  delay(200);

    for (int j=0;j<10;j++)
  {
    analogValue2 += sensor.getDist();
    delay(10);
  }
  analogValue2/=10;
  Serial.print("A2: ");
  Serial.print(analogValue2);
  Serial.println(" cm");

  distanceSensor.startRanging();
  while (!distanceSensor.checkForDataReady())
  {
    delay(1);
  }
  digitalValue2 = distanceSensor.getDistance();
  distanceSensor.clearInterrupt();
  distanceSensor.stopRanging();
  Serial.print("\t\tD2: ");
  Serial.print(digitalValue2);
  if(analogValue1 - analogValue2 > 20 || analogValue2 - analogValue1 > 20 ||
   digitalValue2 - digitalValue1 > 20 || digitalValue1 - digitalValue2 > 20)
{
    digitalWrite(buzzerPin, HIGH);
    if(analogValue1 - analogValue2 > 20 || analogValue2 - analogValue1 > 20)
    {
      Serial.print("\nmove analog ");
```

```
    {
        digitalWrite(buzzerPin, HIGH);
        if(analogValue1 - analogValue2 > 20 || analogValue2 - analogValue1 > 20)
        {
          Serial.print("\nmove analog ");
          digitalWrite(ledPinA, HIGH);
        }
        if(digitalValue2 - digitalValue1 > 20 || digitalValue1 - digitalValue2 > 20)
        {
          Serial.print("\nmove digital ");
          digitalWrite(ledPinD, HIGH);
        }
        delay(100);
        digitalWrite(ledPinA, LOW);
        digitalWrite(ledPinD, LOW);
        digitalWrite(buzzerPin, LOW);
      }
      Serial.println();
      buttonState = digitalRead(buttonPin);
      if( buttonState==HIGH)
        start=0;
      }
   }
}
```

4.GAME END (LOOP FUNCTION):

Purpose:Check for conditions to end the game.

-Check for button press to end the game (start = 0).

5.BUZZER AND LED FEEDBACK:

Purpose: Provide audio and visual feedback during the game.

-Use the buzzer to create sound effects (e.g., countdown, alerts).

-Illuminate LEDs based on detected movement to signal the player. control indicators based on the sensor readings.

THANK YOU!