PROIECT

***PARCAREA SUBTERAN - INTELIGENTĂ***

Disciplina “Măsurări electronice și senzori”

**Realizat de:** Cioban Fabian-Remus

Facultatea de Automatică și Calculatoare

Anul II Grupa 30223

Profesor coordonator : Prof.dr.ing. Radu Munteanu

Cuprins

1. De unde a apărut ideea?
2. Descrierea soluției
3. Domeniu de aplicație
4. Avantaje față de soluțiile existente
5. Cod sursă,imagini exemplificative și o posibilă schemă de montaj realizată în programul circuit.io

# De unde a apărut ideea?

În ziua de azi ,datorită numărului mare de mașini este un chin să găsești un loc de parcare. Așa că ,am zis, de ce să nu proiectez un sistem care te trimite exact într-o parcare liberă. Apoi, vedem pe zi ce trece, numărul mare de infracțiuni ce se abat asupra mașinilor parcate ,neiluminate. Totodată, vedem și

cum în anumite parcări iluminatul este excesiv, fără ca proprietarul să se gândească la economie și resurse. Și nu în ultimul rând ,trebuie să extindem

„viața” mașinii pe timp de iarnă.

# Descrierea soluției

Fiecare participant la trafic care dorește să intre în parcare trebuie să apese pe butonul de la intrare pentru a vedea dacă are sau nu loc liber. În cazul în care se aprinde ledul verde, va apărea traseul iluminat pana la o parcare libera, iar în caz contrar ledul va lumina roșu .Fiecare parcare are un anumit timp de iluminare al traseului până se va stinge. În cazul în care, șoferul va respecta numărul parcării în care trebuie să parcheze, va beneficia de supravegherea mașinii de către un led, pe toată durata opririi. Mai mult decât aceasta, pe timp de iarnă ,parcarea va fi încălzită între anumite valori pentru a extinde durata de viață a mașinii. Dacă șoferul nu va respecta indicația parcării libere, nu va beneficia de nici una din afirmațiile de mai sus .La ieșire fiecare șofer ,va apăsa un buton pentru a reseta în sistem că parcarea a devenit din nou liberă.

# Domeniu de aplicație

* + - Energie
    - Siguranță
    - Întreținere autoturisme
    - Trafic

# Avantaje față de soluțiile existente

-Economisirea energiei electrice

-Economisirea energiei termice

-Decongestionarea traficului

* 1. **Code sursa si imagini exemplificative**

///LCD

#include <LiquidCrystal\_I2C.h>

int totalColumns = 16; ////coloanele lcd int totalRows = 2; ///randuri lcd

LiquidCrystal\_I2C lcd(0x27, totalColumns, totalRows);

///DHT11

#include <DFRobot\_DHT11.h>

DFRobot\_DHT11 DHT;

#define DHT11\_PIN 13

#define LED1 44

#define LED2 45

#define LED3 46

#define LED4 3

#define LED5 28

#define LED6 29

#define LED7 30

#define LED8 31

#define LED9 32

#define LEDtemp 33

const int button = 2; ///buton intrare const int buttoniesire = 4; ///buton iesire

const int redPin = 40; ///RGB rosu const int greenPin = 41; ///RGB verde

///ultrasonic 1 int

const trigPin1 = 6; int

const echoPin1 = 5;

///ultrasonic 2 int

const trigPin2 = 8; int

const echoPin2 = 7;



///ultrasonic 3 int

const trigPin3 = 10; int

const echoPin3 = 9;

///ultrasonic 4 int

const trigPin4 = 12; int

const echoPin4 = 11; const int DUTY = 64;

int ok1 = 0; int ok2 = 0; int ok3 = 0; int ok4 = 0;

void setup() {

lcd.init(); ///initierea lcdului

///1

pinMode(LED1, OUTPUT);

pinMode(trigPin1, OUTPUT); pinMode(echoPin1, INPUT);

///2

pinMode(LED2, OUTPUT);

pinMode(trigPin2, OUTPUT); pinMode(echoPin2, INPUT);

///

pinMode(LED3, OUTPUT);

pinMode(trigPin3, OUTPUT); pinMode(echoPin3, INPUT);

///4

pinMode(LED4, OUTPUT);

pinMode(trigPin4, OUTPUT); pinMode(echoPin4, INPUT);

/// led 5 6 7 8 9 pinMode(LED5, OUTPUT); pinMode(LED6, OUTPUT); pinMode(LED7, OUTPUT); pinMode(LED8, OUTPUT); pinMode(LED9, OUTPUT);

///RGB

pinMode(redPin, OUTPUT);



pinMode(greenPin, OUTPUT);

///temp

pinMode(LEDtemp, OUTPUT);

///buton iesire pinMode(buttoniesire, INPUT\_PULLUP);

///buton intrare pinMode(button, INPUT\_PULLUP);

}

///functie care verifica daca este masina dearupra senzorului 1 (returneaza 0 daca e libera parcarea, 1 daca e ocupata)

int ultrasonic\_1(int u1) { int duration, distance;

digitalWrite(trigPin1, HIGH); delay(1); digitalWrite(trigPin1, LOW);

duration = pulseIn(echoPin1, HIGH); distance = (duration / 2) / 29.1; if (distance <= 3 && distance >= 0)

return 1;

else return 0;

}

///functie care verifica daca este masina dearupra senzorului 2 (returneaza 0 daca e libera parcarea, 1 daca e ocupata)

int ultrasonic\_2(int u2) { int duration, distance;

digitalWrite(trigPin2, HIGH); delay(1); digitalWrite(trigPin2, LOW);

duration = pulseIn(echoPin2, HIGH); distance = (duration / 2) / 29.1; if (distance <= 3 && distance >= 0)

return 1;

else return 0;

}

///functie care verifica daca este masina dearupra senzorului 3 (returneaza 0 daca e libera parcarea, 1 daca e ocupata)

int ultrasonic\_3(int u3) { int duration, distance;

digitalWrite(trigPin3, HIGH); delay(1); digitalWrite(trigPin3, LOW);

duration = pulseIn(echoPin3, HIGH); distance = (duration / 2) / 29.1;



if (distance <= 3 && distance >= 0) return 1;

else return 0;

}

///functie care verifica daca este masina dearupra senzorului 4 (returneaza 0 daca e libera parcarea, 1 daca e ocupata)

int ultrasonic\_4(int u4) { int duration, distance;

digitalWrite(trigPin4, HIGH); delay(1); digitalWrite(trigPin4, LOW);

duration = pulseIn(echoPin4, HIGH); distance = (duration / 2) / 29.1; if (distance <= 3 && distance >= 0)

return 1;

else return 0;

}

void loop() {

int a = 0, b = 0, c = 0, d = 0;

DHT.read(DHT11\_PIN); ///citeste incontinuu temperatura

lcd.backlight(); ///lumina ecrabului lcd lcd.setCursor(0, 0);

lcd.print("Parcarea Fabisor"); lcd.setCursor(0, 1); lcd.print("Temp. "); lcd.print(DHT.temperature);

lcd.print((char) 223); ///caracter pentru cerculetul de la grade lcd.print("C");

delay(1000);

////daca se apasa butoonul de intrare if (digitalRead(button) == LOW) {

////daca e libera parcarea 3 if (ultrasonic\_3(c) == 0) {

///Rgb se face verde digitalWrite(redPin, LOW); digitalWrite(greenPin, HIGH);

/// luminez strada pana la parcare pentru 5 secunde



digitalWrite(LED5, HIGH); digitalWrite(LED6, HIGH); digitalWrite(LED7, HIGH); delay(5000);

/// opresc iluminatul pana la parcare digitalWrite(LED5, LOW); digitalWrite(LED6, LOW); digitalWrite(LED7, LOW);

/// daca s-a parcat masina, pornesc lumina if (ultrasonic\_3(c) == 1) {

digitalWrite(LED3, HIGH); delay(5000); analogWrite(LED3, 80);

ok3 = 1;

}

digitalWrite(greenPin, LOW);

////daca e libera parcarea 2

} else if (ultrasonic\_2(b) == 0) {

///Rgb se face verde digitalWrite(redPin, LOW); digitalWrite(greenPin, HIGH);

/// luminez strada pana la parcare pentru 3.5 secunde digitalWrite(LED5, HIGH);

digitalWrite(LED6, HIGH); delay(3500);

/// opresc iluminatul pana la parcare digitalWrite(LED5, LOW); digitalWrite(LED6, LOW);

/// daca s-a parcat masina, pornesc lumina if (ultrasonic\_2(b) == 1) {

digitalWrite(LED2, HIGH); delay(5000); analogWrite(LED2, 80);

ok2 = 1;

}

digitalWrite(greenPin, LOW);

////daca e libera parcarea 1

} else if (ultrasonic\_1(a) == 0) {

///Rgb se face verde digitalWrite(redPin, LOW); digitalWrite(greenPin, HIGH);

/// luminez strada pana la parcare pentru 2 secunde



digitalWrite(LED5, HIGH); delay(2500);

/// opresc iluminatul pana la parcare digitalWrite(LED5, LOW);

/// daca s-a parcat masina, pornesc lumina if (ultrasonic\_1(a) == 1) {

digitalWrite(LED1, HIGH);

delay(5000); analogWrite(LED1, 80);

ok1 = 1;

}

digitalWrite(greenPin, LOW);

////daca e libera parcarea 4

} else if (ultrasonic\_4(d) == 0) {

digitalWrite(redPin, LOW); digitalWrite(greenPin, HIGH);

/// luminez strada pana la parcare pentru secunde digitalWrite(LED5, HIGH);

digitalWrite(LED6, HIGH); digitalWrite(LED7, HIGH); digitalWrite(LED8, HIGH); delay(6000);

/// opresc iluminatul pana la parcare digitalWrite(LED5, LOW); digitalWrite(LED6, LOW); digitalWrite(LED7, LOW); digitalWrite(LED8, LOW);

/// daca s-a parcat masina, pornesc lumina if (ultrasonic\_4(d) == 1) {

digitalWrite(LED4, HIGH); delay(5000); analogWrite(LED4, 80);

ok4 = 1;

}

digitalWrite(greenPin, LOW);

////daca NU e libera nici o parcare

} else if (ultrasonic\_3(c) == 1 && ultrasonic\_2(b) == 1 && ultrasonic\_1(a) == 1 && ultrasonic\_4(d) == 1) {

digitalWrite(redPin, HIGH); digitalWrite(greenPin, LOW);



digitalWrite(LED5, LOW); digitalWrite(LED6, LOW); digitalWrite(LED7, LOW); digitalWrite(LED8, LOW); delay(5000); digitalWrite(redPin, LOW);

}

}

///daca e sub 27 grade

if (DHT.temperature < 25) {

////si daca exita macar 1 masina in parcare

if (ok1 == 1 || ok2 == 1 || ok3 == 1 || ok4 == 1)

///se porneste incalzirea digitalWrite(LEDtemp, HIGH);

} else if (DHT.temperature > 29)

///altfel se opreste digitalWrite(LEDtemp, LOW);

///la buton iesire verificam care parcare a fost eliberata si stingem becul parcarii if (digitalRead(buttoniesire) == LOW) {

if (ultrasonic\_3(c) == 0) { digitalWrite(LED3, LOW); ok3 = 0;

}

if (ultrasonic\_2(b) == 0) { digitalWrite(LED2, LOW); ok2 = 0;

}

if (ultrasonic\_1(a) == 0) { digitalWrite(LED1, LOW); ok1 = 0;

}

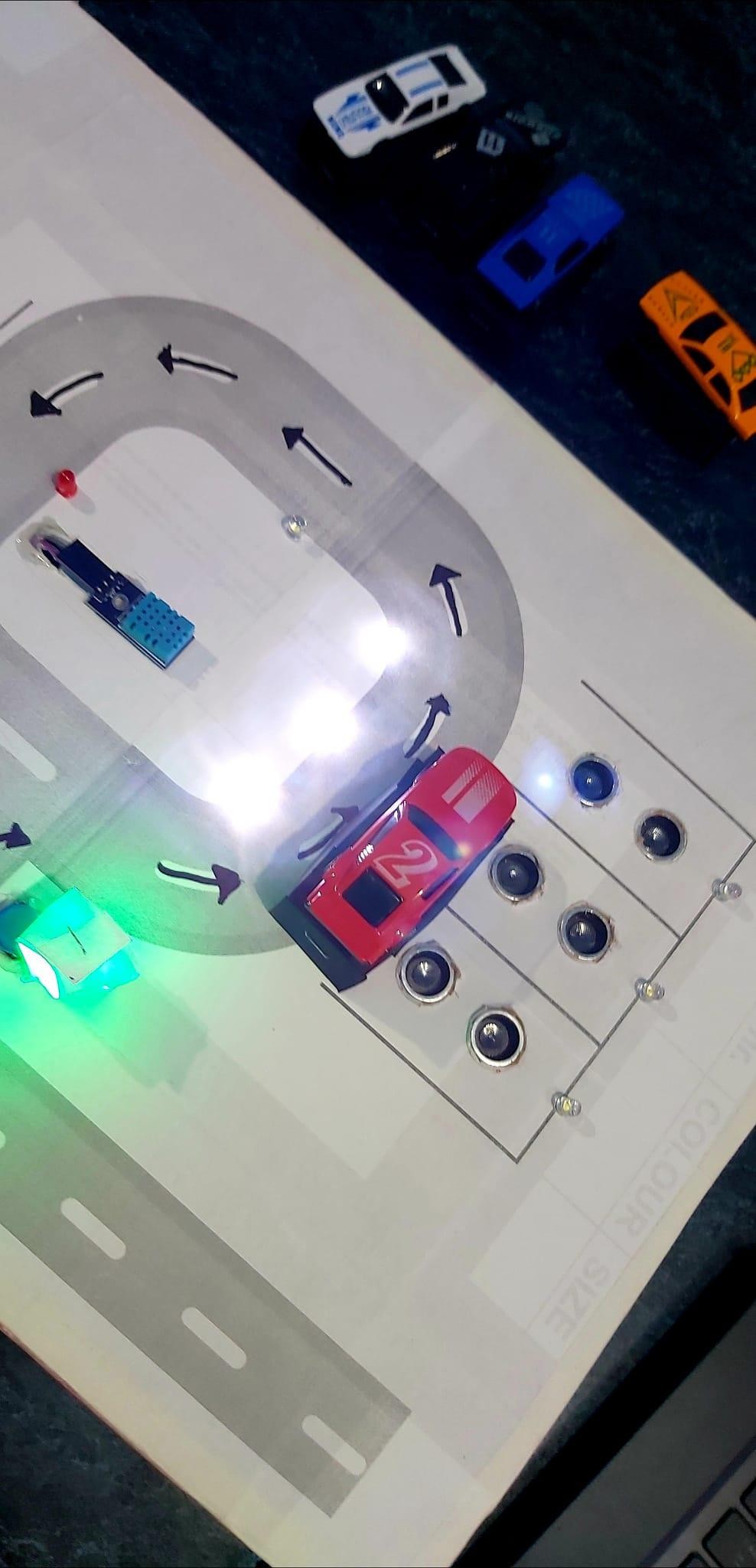
if (ultrasonic\_4(d) == 0) { digitalWrite(LED4, LOW); ok4 = 0;

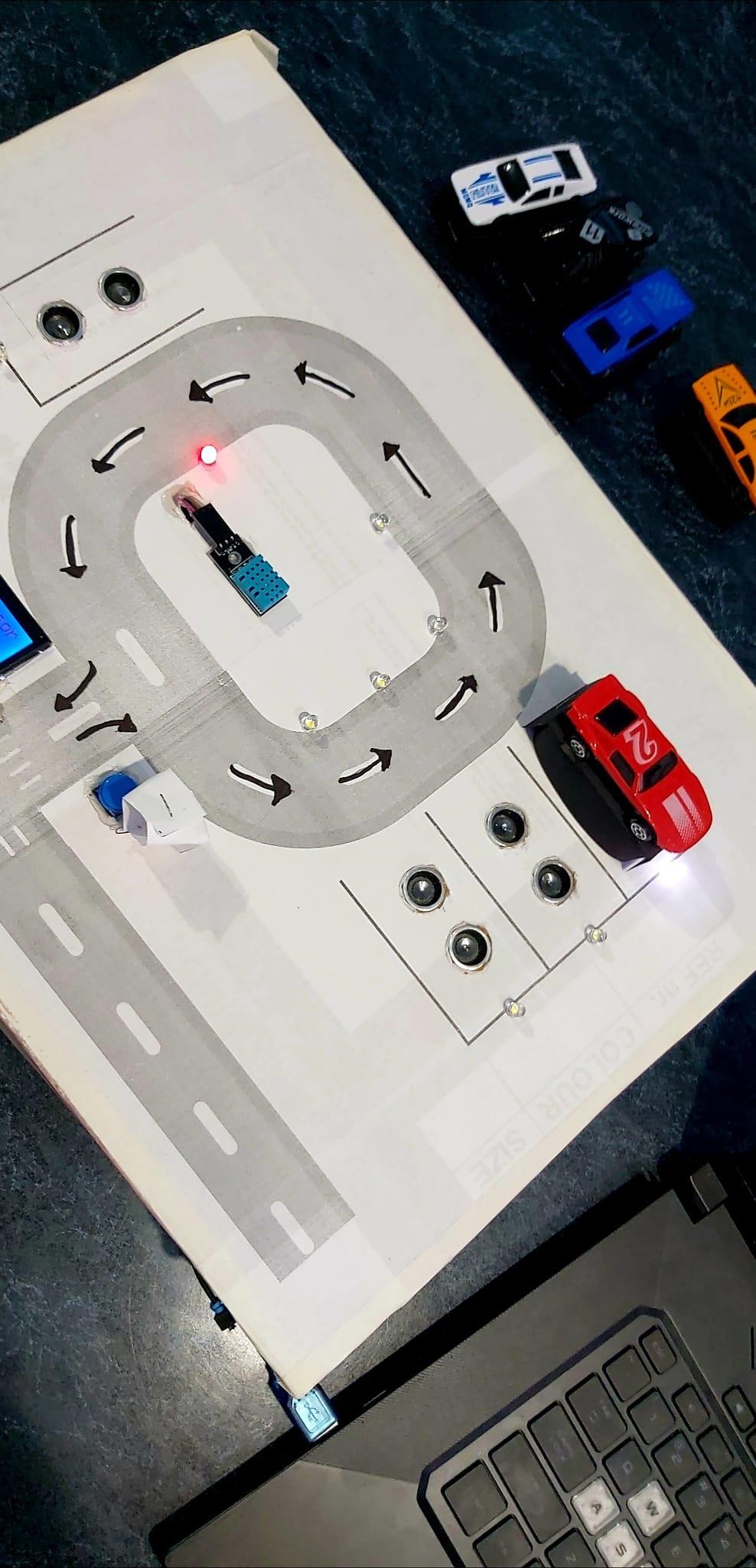
}

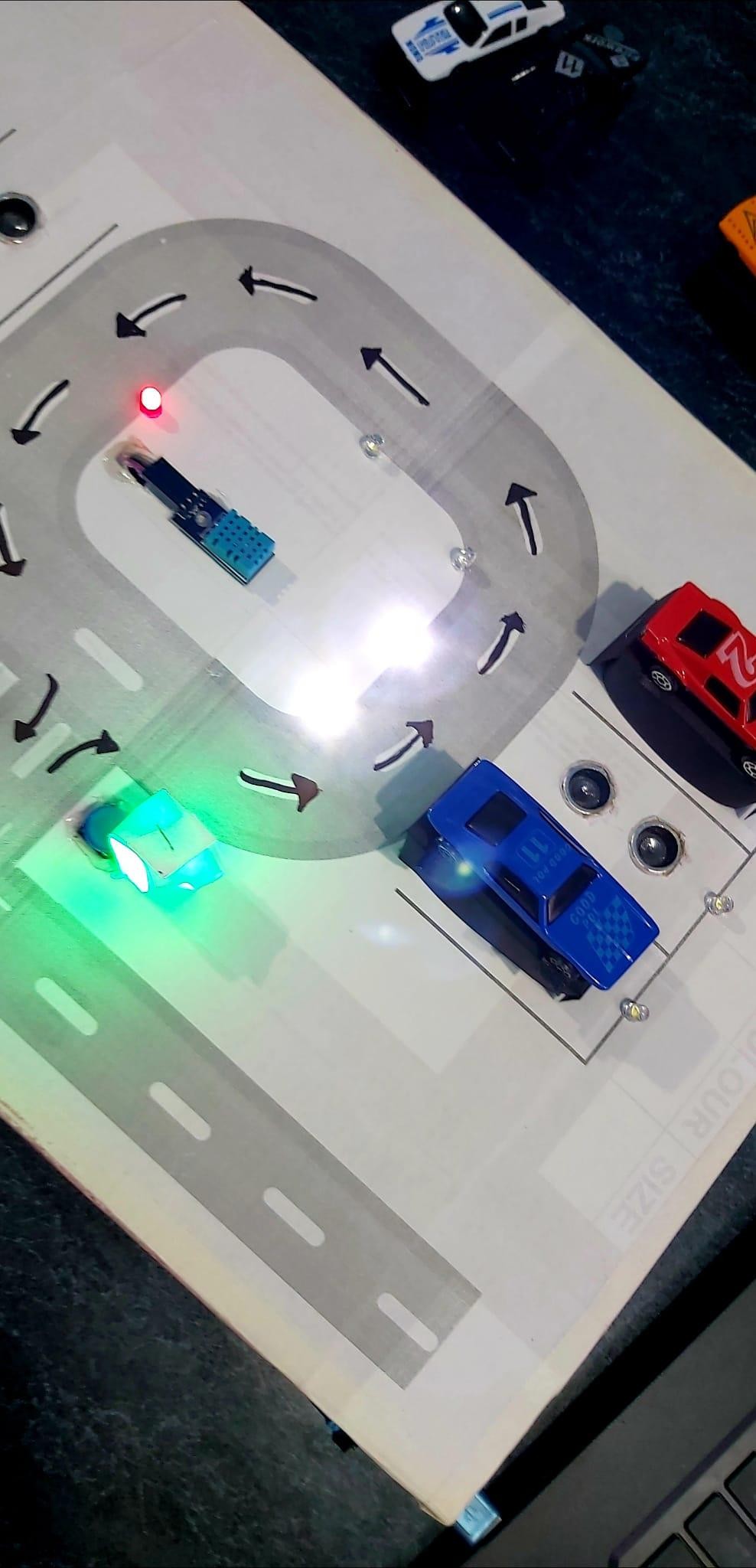
}

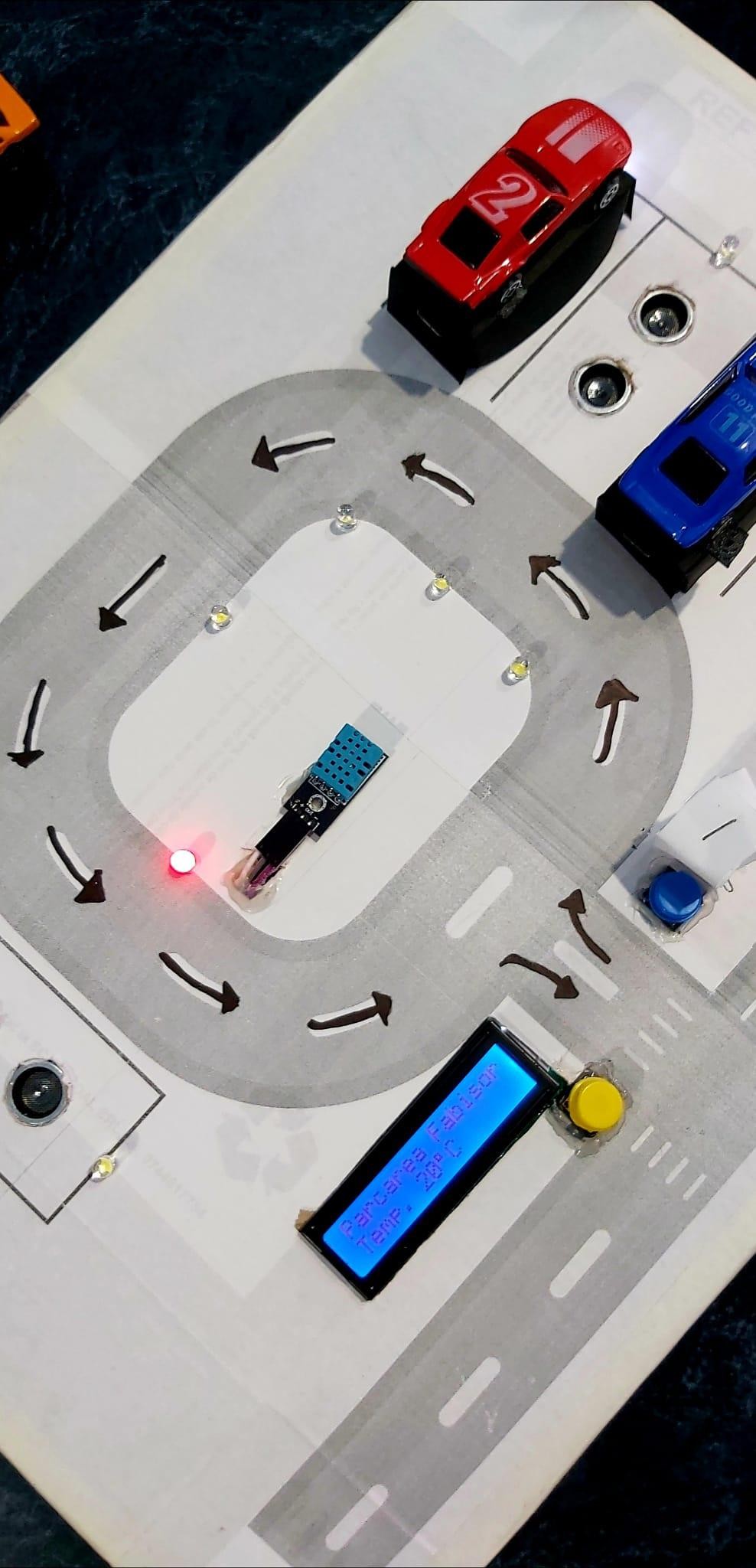
if (ok1 == 0 && ok2 == 0 && ok3 == 0 && ok4 == 0) digitalWrite(LEDtemp, LOW);

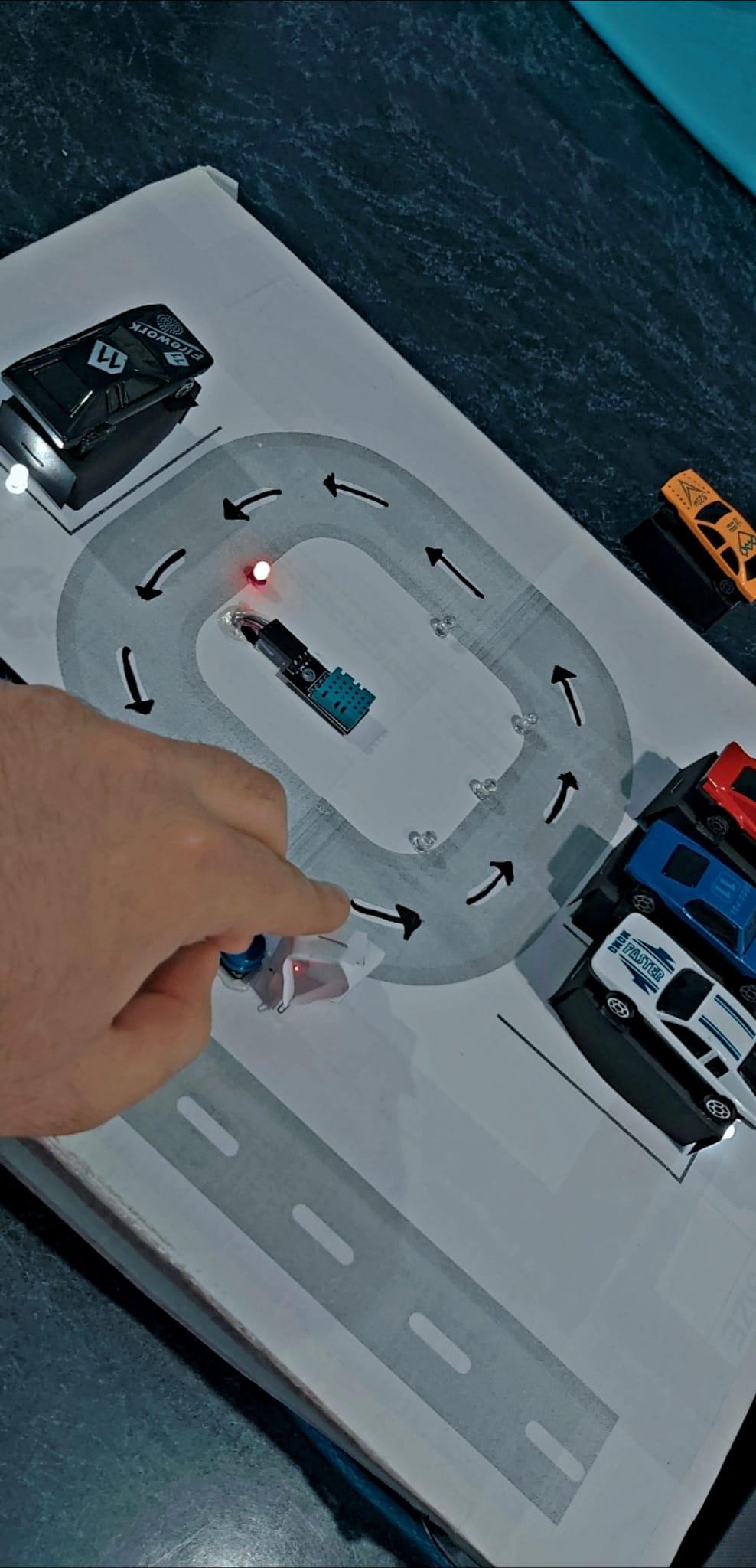
}

Traseul până la prima parcare liberă

Supraveghere mașină dacă a fost parcată corect

 Parcarea incorectă a autoturismului, luând în calcul regulile parcării

Facilitatea de încălzire a parcării dacă este minim o mașină parcată corect

Parcarea este complet ocupată

Ieșirea unei mașini din parcare

A picture containing text

Description automatically generated

Diagram, schematic

Description automatically generated