Fire Alarm - Selea Tudor-Octavian 332CA

Introducere

Ideea proiectului constă în realizarea unei alarme de incendiu. Aceasta va prelua date în timp real despre temperatura și cantitatea de fum din încăperea în care este plasată alarma și, în cazul în care temperatura și cantitatea de fum depășesc limitele setate, alarma va emite un semnal sonor, un LED roșu se va aprinde, mesajul de “Pericol incendiu!” va fi afișat pe un LCD. Pentru a opri alarma, va fi necesară introducerea unui cod PIN setat de programator. Când alarma nu este declanșată, temperatura camerei va fi afisata pe LCD.

Descriere generală

Senzorii de temperatură și fum vor prelua temperatura, respectiv cantitatea de fum din încăpere. Aceste date vor fi citite de o plăcuță Arduino, care va activa piezo-rezistorul, LED-ul și va transmite mesajul “Pericol incendiu!” către ecranul LCD dacă valoarile datelor citite de Arduino sunt mai mari decat limitele setate de programator. În caz contrar, plăcuța Arduino va afișa datele citite de la senzorul de temperatură și le va afișa ecranul LCD. În momentul pornirii alarmei, prima placuță Arduino va înștiința a doua plăcuță Arduino, care va avea ca input o tastatură, că alarma a fost pornită. Pentru a dezactiva alarma, utilizatorul va trebui să intrucă un cod PIN la tastatura care să corespundă cu codul PIN setat pe a doua plăcuță. În momentul în care utilizatorul introduce codul bun, a doua plăcuță îi dă acordul primei plăcuțe să dezactiveze alarma.

Schema bloc

Hardware Design

Piesele folosite în cadrul proiectului sunt:
- 2 x Arduino Uno ATmega328P
- Senzor de temperatura TMP36
- Breadboard
- Fire
- Senzor de fum MQ-7
- Display LCD1602 I2C
- Buzzer pasiv 5V
- Led de 5 mm
- Keypad
- Rezistenta 220Ω

Schema electrica

Export to PDF

Software Design

Implementarea software a proiectului poate fi reprezentata prin urmatoarea diagrama de stari:

Pentru dezvoltarea proiectului am folosit **Arduino IDE** si librariile:

- **LiquidCrystal_I2C** - folosirea display-ului LCD 1602 I2C
- **Wire.h** - comunicarea cu LCD-ul 1602 I2C
- **Keypad.h** - folosirea keypad-ului

Implementarea software este formata din 2 fisiere: sender_pm.ino si receiver_pm.ino .

Sender_pm.ino

Acest fisier este incarcat pe placuta care controleaza:

- **Buzzer-ul**
- **LED-ul**
- **LCD-ul**
- **Senzorul de gaz**
- **Senzorul de temperatura**

Arduino va transmite date LCD-ul prin protocolul I2C, astfel ca e nevoie de adresa LCD-ului (0x27 in cazul de fata) in momentul initierii conexiunii dintre Arduino si LCD. Buzzer-ul este conectat la pinul 7 al placutei Arduino, iar LED-ul la pinul 13. Variabila “alarm_trigger”, care devine 1 in cazul aparitiei unui incendiu, este initial setata initial pe 0.

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

LiquidCrystal_I2C lcd(0x27,16,2);
```

```
// Temperatura in grade Celsius
double temp;

// Temperatura in Volti
double temp_voltage;

// Output-ul senzorului de temperatura
int read_temp;

int LED = 13;
int buzzer = 7;

// Output-ul senzorului de gaz
int read_gas;

// Daca alarm_trigger = 1, alarma a fost declansata
char alarm_trigger = 0;
```

In setup, A0 si A1 sunt setati ca pini de input, A0 primind concentratia de gaz de la senzorul de gaz, iar A1 primind temperatura de la senzorul de temperatura. Buzzer-ul este setat pe HIGH initial deoarece el este activ pe LOW, iar acesta trebuie sa fie activ doar in momentul declansarii alarmei. Cand alarma este pornita, mesajul “Loading Fire Alarm...” este afisat pe LCD timp de 5 secunde.

```
void setup()
{
  pinMode(A0, INPUT);
  pinMode(A1, INPUT);
  pinMode(LED, OUTPUT);
  pinMode(buzzer, OUTPUT);

  // LED-ul este setat initial pe LOW
  digitalWrite(LED, LOW);

  // Buzzer-ul este setat initial pe HIGH deoarece el este activ pe LOW
  digitalWrite(buzzer, HIGH);

  // Initializarea LCD-ului
  lcd.begin(16, 2);
  lcd.init();
  lcd.backlight();

  // Initializarea comunicarii Seriale si a monitorului Serial
  Serial.begin(9600);

  // Cand alarma este pornita, mesajul "Loading Fire Alarm..." este
  // afisat pe LCD
  lcd.setCursor(0, 0);
  lcd.print("Loading Fire");
  lcd.setCursor(0, 1);
  lcd.print("Alarm...");
  delay(5000);
```

```
  lcd.clear();
}
```

Senzorul de temperatura TMP36 va trimite placutei Arduino o valoare intre 0 si 1024, care este ulterior convertita in grade Celsius.

```
// Arduino citeste output-ul senzorului de temperatura TMP36
  read_temp = analogRead(A1);

  // Convertim tensiunea citita din valoarea numerica ADC in volti
  temp_voltage = (double)read_temp / 1024;
  temp_voltage *= 5;

  // Convertim temperatura obtinuta din volti in grade celsius
  // Obs! Offset-ul de 0.5 ajuta la citirea temperaturilor negative
  temp = (temp_voltage - 0.5) * 100;
```

Daca temperatura inregistrata de senzor depaseste 70 de grade Celsius, iar senzorul de gaz detecteaza o densitate a monoxidului de carbon mai mare de 150 ppm, alarma se va declansa prin setarea LED-ului pe HIGH, buzzer-ului pe LOW si prin afisarea mesajului “Pericol incendiu!!!” pe LCD.

Totodata, variabila “alarm_trigger” va fi actualizata la valoarea 1 si trimisa celei de a doua placute Arduino pentru a activa keypad-ul.

Programul nu-si va continua executia pana cand “alarm_trigger” nu va fi actualizat la 0 de catre a doua placuta Arduino, confirmand, astfel, dezactivarea cu succes a alarmei. Dupa dezactivarea alarmei, mesajul “Alarma oprita!” va fi afisat timp de 5 secunde, dupa care programul va trece la urmatoarea iteratie a loop-ului.

```
// Daca temperatura si concentratia de gaz depasesc valorile limita,
  // LED-ul va fi setat pe HIGH, buzzer-ul va fi setat pe LOW, iar pe
ecranul
  // LCD va fi afisat un mesaj care indica pericolul aparitiei unui incendiu
  if (read_gas >= 150 && temp >= 70)
  {
    alarm_trigger = 1;
    digitalWrite(LED, HIGH);
    digitalWrite(buzzer, LOW);
    lcd.setCursor(0, 0);
    lcd.print("Pericol incendiu");
    lcd.setCursor(0, 1);
    lcd.print("!!!!!!!!!!!!!!!!");

    // Aceasta placuta Arduino va trimite celei de a doua placute Arduino
    // variabila "alarm_trigger", care activeaza tastatura numerica
(conectata
    // la a doua placuta Arduino) atunci cand "alarm_trigger" = 1;
    Serial.write(alarm_trigger);

    // Placuta Arduino asteapta de la a doua placuta confirmarea ca alarma a
    // fost dezactivata (alarm_trigger = 0)
```

```
    while (1) {
      if (Serial.available()) {
        alarm_trigger = Serial.read();
        if (alarm_trigger == 0) {
          break;
        }
      }
    }

    // In momentul in care alarma a fost oprita, LED-ul este setat pe LOW,
buzzer-ul
    // este dezactivat prin setarea pe HIGH, iar mesajul "Alarma oprita!"
este afisat
    // pe ecranul LCD timp de 5 secunde, dupa care alarma va fi reactivata
    lcd.clear();
    digitalWrite(LED, LOW);
    digitalWrite(buzzer, HIGH);
    lcd.setCursor(0, 0);
    lcd.print("Alarma oprita!");
    delay(5000);
    lcd.clear();

  }
```

Daca alarma nu este activata, ecranul LCD va afisa temperatura camerei, actualizata la fiecare secunda.

```
    digitalWrite(LED, LOW);
    digitalWrite(buzzer, HIGH);
    lcd.setCursor(0, 0);
    lcd.print("Temperatura:");
    lcd.setCursor(0,1);
    String output = String(temp) + String((char)0xDF) + String("C");
    lcd.print(output);
```

Receiver_pm.ino

Acest fisier este incarcat pe placuta care controleaza:

- **Keypad-ul**

In “alarm_trigger” va fi retinuta variabila trimisa de prima placuta Arduino, care va semnala declansarea alarmei. Matricea “keys” retine simbolurile keypad-ului, ROWS si COLS retin dimensiunile keypad-ului, iar rowPins si colPins retin pinii alocati pentru fiecare linie si coloana din keypad. Pe baza acestor variabile se va face maparea keypad-ului la Arduino.

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

```
#define PASSWORD_LENGTH 10

char alarm_trigger;

// Parola introdusa de utilizator
char password[PASSWORD_LENGTH];

// Parola corecta
char correct_password[PASSWORD_LENGTH]="13468*#AD";
char count = 0;

// Keypad-ul are 4 linii si 4 coloane
const byte ROWS = 4;
const byte COLS = 4;

// Simbolurile keypad-ului
char keys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};

// Pinii arduino corespunzatori tuturor coloanelor si liniilor
// keypad-ului
byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};

// Vom genera un keypad pe baza dimensiunilor si a simbolurilor setate
// de programator
Keypad customKeypad = Keypad(makeKeymap(keys), rowPins, colPins, ROWS, COLS);
```

Daca nu exista conectivitate cu prima placuta Arduino, nu pot fi citite date pe portul serial.

Daca exista conectivate, se va citi semnalul de alarma trimis de catre prima placuta Arduino in “alarm_trigger”. Daca acesta este 1, alarma a fost declansata, astfel ca poate fi introdus codul pentru dezactivarea acesteia. Fiecare tasta apasata de utilizator va fi retinuta in variabila “customKey” si va fi adaugata in array-ul “password”, care reprezinta parola introdusa de utilizator. Daca dimensiunea parolei introduse de utilizator este egala cu dimensiunea parolei corecte, se vor compara cele 2 parole. In caz de egalitate (utilizatorul a introdus parola corecta), prima placuta Arduino va fi instiintata de catre a doua placuta Arduino (placuta curenta), ca poate sa dezactiveze alarma, iar keypad-ul va fi dezactivat pana la urmatoarea declansarea a alarmei. In caz contrar, se va reincerca introducerea parolei corecte.

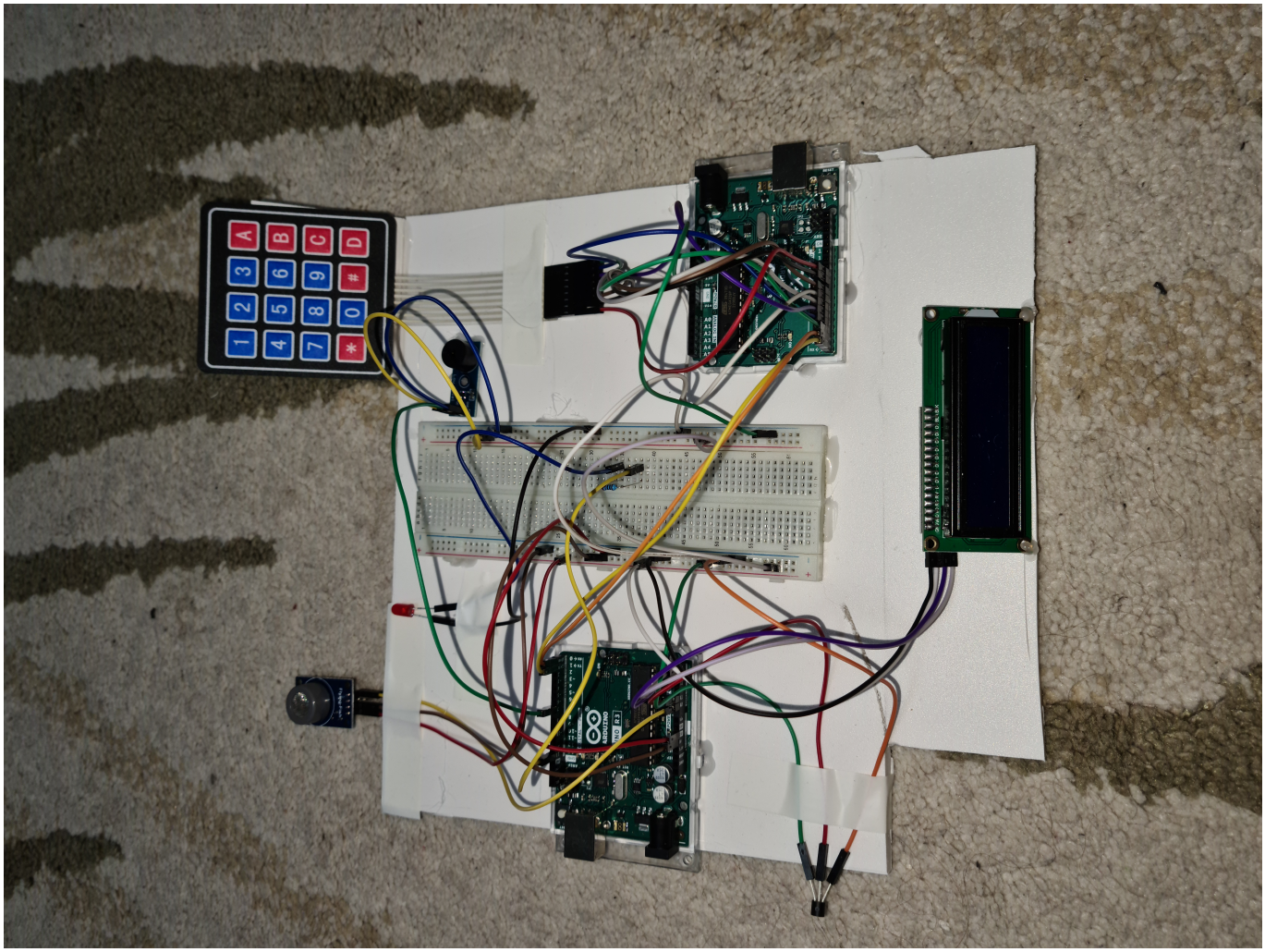
```
void loop(){

  // Daca nu exista conectivitate cu prima placuta Arduino
  // nu pot fi citite date pe portul serial
  if (Serial.available()) {
    Serial.print("In serial loop...");
    Serial.println();
```

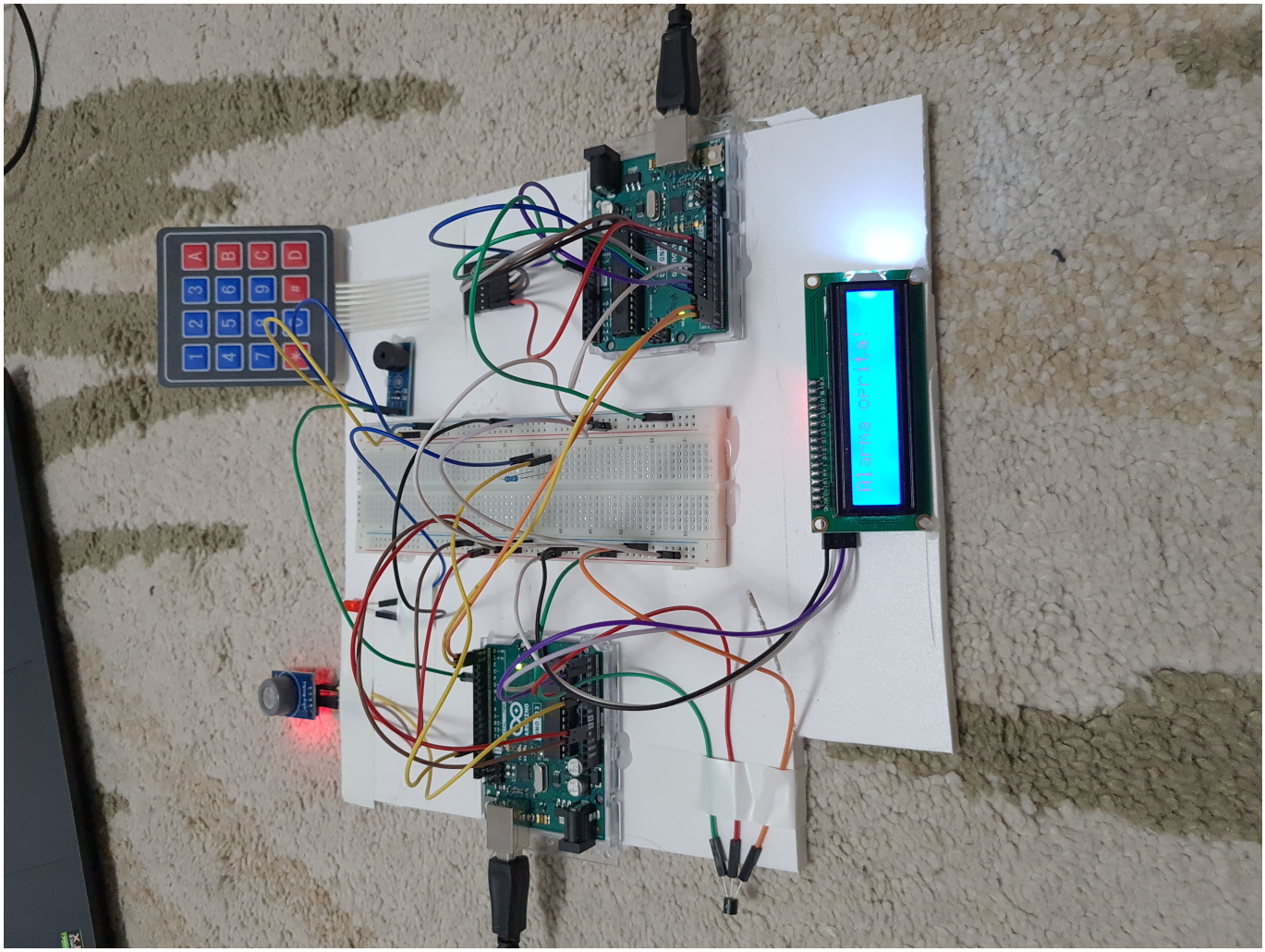
```
// Variabila care semnaleaza declansarea alarmei
alarm_trigger = Serial.read();

if (alarm_trigger == 1) {
  Serial.print("Alarm triggered...");
  Serial.println();

  // Daca alarma a fost declansata, se va incerca
  // introducerea parolei pana cand este cea corecta
  while(1) {

    // Tasta apasata de utilizator va fi intoarsa in
    // variabila "customKey"
    char customKey = customKeypad.getKey();

    if(customKey){

      // Caracterul corespunzator tastei apasate de
      // utilizator va fi adaugat in parola introdusa
      // de utilizator
      password[count]=customKey;
      Serial.print(password[count]);
      count++;
    }

    // In momentul in care lungimea parolei introduse
    // de utilizator este egala cu lungimea parolei
    // corecte, se va verifica daca parola introdusa
    // de utilizator este corecta
    if(count == PASSWORD_LENGTH - 1){
      Serial.println(" ");

      if(strcmp(password,correct_password) == 0){

        // Dupa verificarea parolei introduse de
        // utilizator, aceasta va fi stearsa si pregatita
        // pentru reintroducerea parolei, cand va fi cazul
        while(count !=0) {
          password[count]=0;
          count--;
        }

        // Daca parola introdusa de utilizator este corecta,
        // prima placuta Arduino va fi instiintata de catre
        // a doua placuta Arduino (placuta curenta), ca poate
        // sa dezactiveze alarma
        Serial.println("Correct");
        alarm_trigger = 0;
        Serial.write(alarm_trigger);
        break;
```

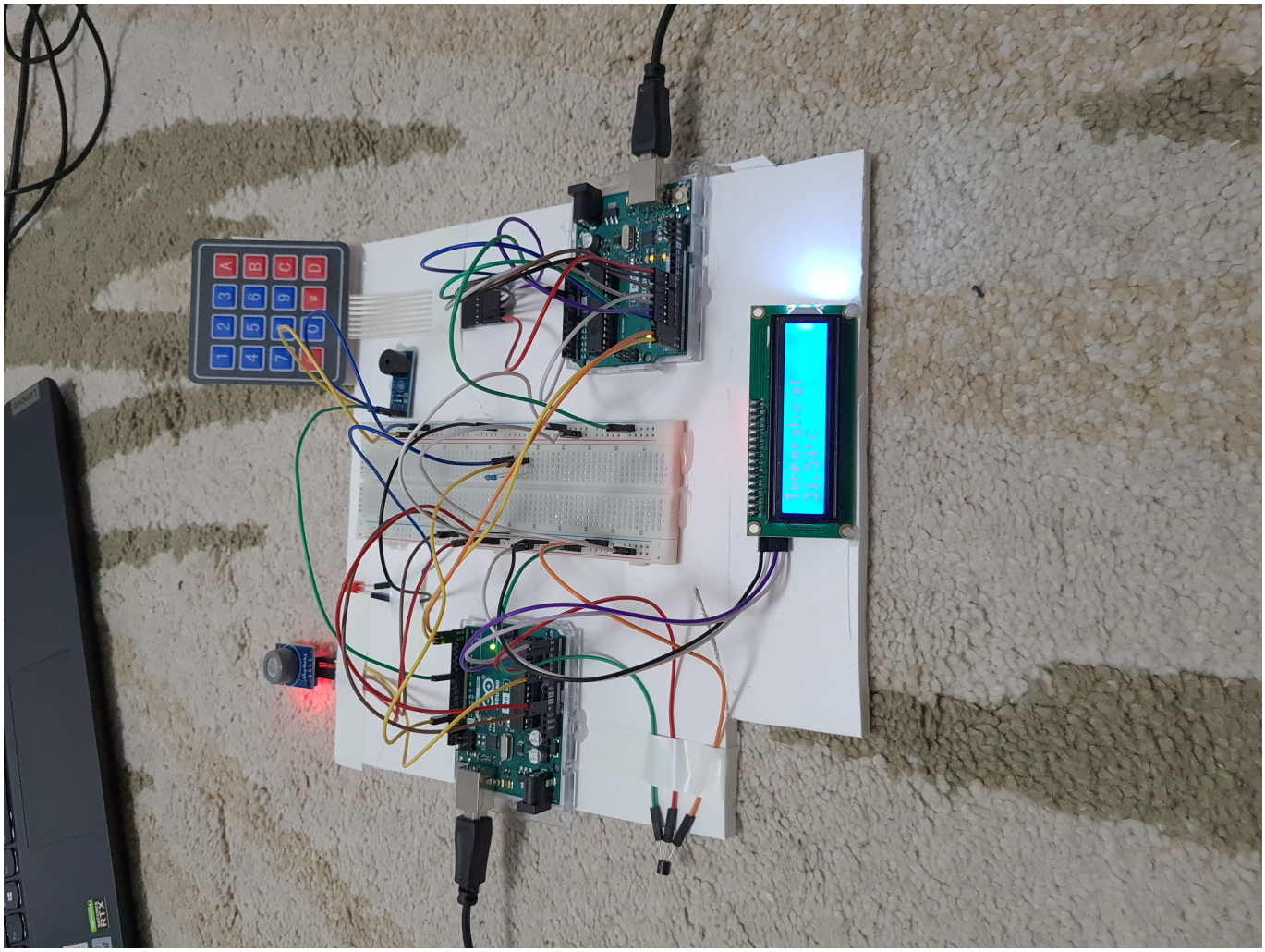
```
          } else{

            // Daca parola introdusa de utilizator este incorecta,
            // se va reincerca introducerea parolei
            Serial.println("Incorrect");
            alarm_trigger = 1;
          }

          while(count !=0){
              password[count]=0;
              count--;
          }
        }
      }
    }
  }
}
```

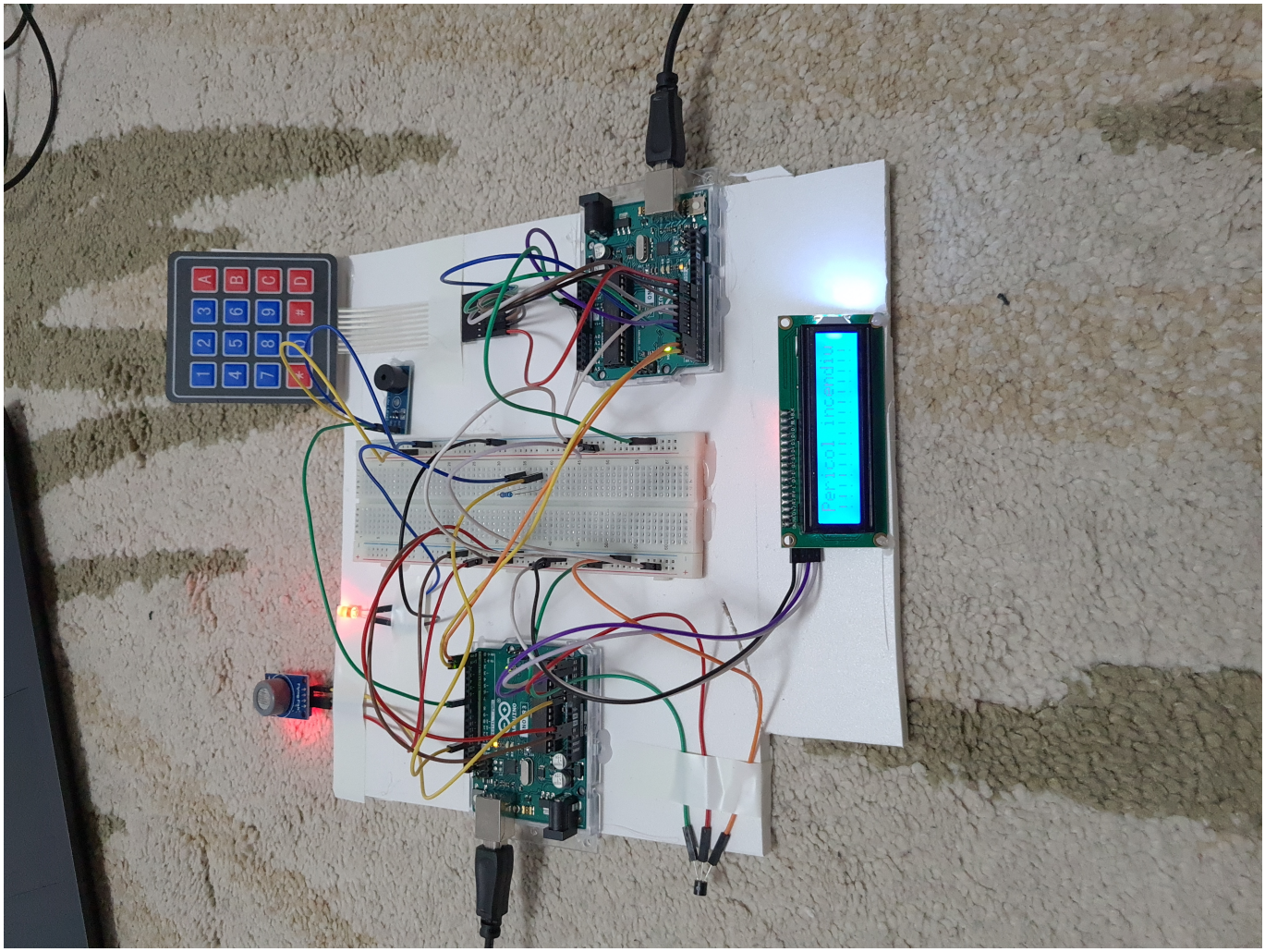
Dupa introducerea unei parole de catre utilizator, aceasta va fi stearsa!

Rezultate obtinute


A B C D
3 6 9 #
2 5 8 0
1 4 7 *


Alarma aprila


Temperature
31 Celsius



Concluzii

Mi-a placut sa lucrez la acest proiect intrucat am putut sa pun in practica notiunile invatate la laboratoare pentru a realiza un lucru cu aplicatie in viata. Mi-a placut, in special, sa lucrez cu componentele proiectului si sa observ cum se leaga si cum comunica intre ele.

Download

selea_tudor_octavian_-_332ca_-_proiect_pm.zip

Resurse

- https://ocw.cs.pub.ro/courses/pm/lab/lab4-2022
- https://ocw.cs.pub.ro/courses/pm/lab/lab1-2022
- https://ocw.cs.pub.ro/courses/pm/lab/lab6-2022

- https://www.youtube.com/watch?v=sPhcOm3FdOQ&t=254s&ab_channel=DomingoMartinez
- https://linuxhint.com/serial-uart-communication-between-two-arduino/
- Proiect AD - Selea Tudor-Octavian (vezi sectiunea Downloads)

From:
http://ocw.cs.pub.ro/courses/ - **CS Open CourseWare**

Permanent link:
http://ocw.cs.pub.ro/courses/pm/prj2023/vstoica/fire_alarm

Last update: **2023/05/29 17:06**