

Sistem de alarma

Nume: Soare Mihai-Daniel

Grupa: 331CA

Introducere

Proiectul consta in realizarea unui sistem de alarma pentru o locuinta. Scopul acestei alarme este de a instiinta clientul daca o persoana neautorizata patrunde in locuinta clientului si il informeaza instant.

Clientului ii este disponibil sa isi securizeze locuinta cu o anumita parola care trebuie introdusa de fiecare data cand doreste sa intre in locuinta sau sa aiba cardul rfid corespunzator alarmei la indemana.

Se doreste a fi o alarma cat mai veritabila, deci se va folosi un modul de comunicatie pentru a-l instiinta pe client.

De asemenea, o alarma sonora va porni cand persoana neautorizata intra in locuinta.

Descriere generală

In primul rand, clientul va interactiona cu o tastatura si cu un display lcd pentru setarea parolei pe care orice persoana care trece de pragul unui senzor trebuie sa o introduca. Pe langa introducerea parolei, clientul poate sa foloseasca un card rfid pentru dezactivarea alarmei.

Introducerea parolei sau folosirea cardului va duce la dezactivarea alarmei pentru o anumita perioada de timp, cat clientul intra in locuinta.

Daca o persoana neautorizata incearca sa treaca de prag, va porni un sunet de alarma, se vor aprinde niste becuri si un e-mail se va trimite catre client.

Sistemul va fi conectat la wi-fi printr-un modul precum ESP8266, iar toate setarile legate de internet vor fi realizate de persoana care are acces la sistemul de alarma (in cazul de fata detinatorul proiectului).

Pentru a se trimite informatia catre client prin e-mail, acesta isi poate seta un e-mail, altfel informatia va fi trimisa catre compania care a realizat sistemul de alarma. (in cazul de fata detinatorul proiectului)

Schema bloc



Hardware Design

Lista piese:

- Placa Arduino
- Breadboard
- Ultrasonic Sensor
- Display LCD
- LED-uri
- Tastatura
- Modul wi-fi
- Modul rfid
- Fire de legatura
- Buzzer

In urma realizarii schemei electrice, am descoperit ca era imposibil sa realizez proiectul daca adaugam si tastatura, astfel am ramas la toate celelalte mai putin tastatura.

Schemă electrică



Schemă fizică



Software Design

Inca de la etapa de hardware, am facut un research mai amanuntit asupra a ce vreau sa ajunga in software design, asadar am cautat biblioteci utile pentru fiecare componenta aleasa, cat si m-am gandit cum ar trebui implementat sistemul de trimitere a email-urilor.

Bibliotecile folosite sunt:

- NewPing → folosita la senzorul de miscare
- SPI si MFRC522 → folosite la cititorul de carduri RFID
- LiquidCrystal_I2C → folosita la ecranul LCD
- SoftwareSerial → folosita la modulul wi-fi esp8266

Setup

Pe partea de setup avem urmatoarea functie:

```
void setup()
{
    // buzzer setup
    pinMode(BUZZER_PIN, OUTPUT);
    analogWrite(BUZZER_PIN, 255);

    Serial.begin(9600);
    delay(100);

    wifi_setup();

    // rfid setup
    SPI.begin();
    mfrc522.PCD_Init();
    mfrc522.PCD_DumpVersionToSerial();

    delay(100);

    // initialize the LCD
    lcd.begin();

    // turn on the backlight and print a message.
    lcd.backlight();
    lcd.print("Home security");
}
```

Setup ESP8266

Pentru setup-ul modulului wi-fi am realizat functia:

```
// helper function used to send an AT command to esp module
// and run it there
void sendATCommand(const char* command, int timeout = 1000) {
    esp8266.println(command);
    delay(timeout);
    while (esp8266.available()) {
        char c = esp8266.read();
        Serial.write(c);
    }
    delay(10); // Delay between commands
}
```

```
}

void wifi_setup()
{
    esp8266.begin(9600);
    delay(2000);
    sendATCommand("AT");
    delay(5000);
    sendATCommand("AT+CWMODE=3");
    delay(5000);
    // change SSID and PASSWORD
    sendATCommand("AT+CWMJAP=\"SSID\", \"PASS\"");
    delay(9000); // Longer delay after connecting to WiFi
    sendATCommand("AT+CIFSR");
    delay(100);
}
```

Aceasta componenta a proiectului este una destul de complex, petrecand o mare parte a timpului intelegand sintaxa esp8266 AT care realizeaza diferite actiuni pe modulul wi-fi.

Codul anterior face ca placuta Arduino sa se conecteze la o retea wi-fi prin intermediul modulului ESP8266.

Logica principala

Logica principala a codului rulat pe Arduino este destul de simpla:

- se calculeaza distanta pe raza senzorului de miscare
- se verifica daca cumva a fost introdus cardul
- se verifica daca exista o persoana care a trecut prin fata senzorului de miscare si nu a folosit cardul.
- daca da, suna o alarma, se modifica ecranul si se trimite o cerere catre serverul deschis care se ocupa de trimiterea mail-urilor
- altfel se verifica daca nu cumva cartela a fost introdusa si se reseteaza tot sistemul.

```
void loop()
{
    int distance = sonar.ping_cm();
    Serial.print("Distance is: ");
    Serial.println(distance);
    delay(100);

    check_card();
    if ((distance > 5 && distance < 40) && inserted_card == false && detected == false) {
        // trigger LCD for intruder
        lcd.clear();
        lcd.print("INTRUDER");

        // trigger alarm
        Serial.println("DANGER!");
        trigger_alarm();
    }
}
```

```
// wait for alarm to finish
delay(3000);

// connect to the TCP server and sends an email to owner
sendATCommand("AT+CIPSTART=\"TCP\", \"IP_ADDRESS\", 8080");
delay(1000);

// clear detection
detected = false;

// clear lcd detection
lcd.clear();
lcd.print("Home security");
}

// close access after 3 secs
if (inserted_card == true) {
    lcd.clear();
    lcd.print("Welcome home!");

    // set time for entry in the house
    delay(15000);

    // closing the door so resetting the alarm
    inserted_card = false;
    lcd.clear();
    lcd.print("Home security");
}
}
```

Pe langa functia principala de loop, am realizat inca 2 functii utile in urmatoarele situatii:

→ suna alarma cand exista o persoana neautorizata care intra in locuinta

```
void trigger_alarm()
{
    detected = true;
    for (int i = 0; i < 2000; i++) {
        analogWrite(BUZZER_PIN, 222);
        delay(2);
    }
    delay(100);
    analogWrite(BUZZER_PIN, 255);
}
```

→ realizeaza verificarea existentei a cardului care opreste alarma temporar in preajma cititorului de carduri rfid

```
void check_card()
{
    if (mfrc522.PICC_IsNewCardPresent()) {
```

```
if (mfrc522.PICC_ReadCardSerial()) {
  Serial.print("Card detected. UID: ");
  Serial.println(mfrc522.uid.size);
  for (byte i = 0; i < mfrc522.uid.size; i++) {
    Serial.print(mfrc522.uid.uidByte[i] < 0x10 ? " 0" : " ");
    Serial.print(mfrc522.uid.uidByte[i]);
  }
  // check if the card of the owner was inserted
  if (mfrc522.uid.uidByte[0] == 187 && mfrc522.uid.uidByte[1] == 110
      && mfrc522.uid.uidByte[2] == 242 && mfrc522.uid.uidByte[3] == 90)
  {
    Serial.println();
    Serial.println("CARD INSERTED");
    inserted_card = true;
  }
  Serial.println();
  mfrc522.PICC_HaltA();
}
}
delay(100); // Adjust the delay as needed
}
```

Server TCP

Acum pe partea de server remote, am creat un server in python care poate fi rulat pe mai multe arhitecturi, in situatia actuala doar pe local (poate fi mutat pe remote), in care se realizeaza o conexiune TCP intre placuta Arduino si server.

Serverul odata ce primeste o noua conexiune va trimite un email catre ownerul sistemului de alarma.

Acesta este codul serverului si este destul de clar explicat prin comentariile lasate.

```
import socket
import smtplib
from email.mime.text import MIMEText

def handle_client(client_socket):
    # Print a message when a client connects
    print("Arduino connected")

    # Send a message to the Arduino
    client_socket.send(b"Hello, Arduino!")

    send_email()

    # Close the client socket
    client_socket.close()

def send_email():
    smtp_host = 'mail.smtp2go.com'
    smtp_port = 2525
```

```
sender_email = 'mihai.orange7@gmail.com'
recipient_email = 'mihai_daniel.soare@stud.acs.upb.ro'
smtp_username = 'USER'
smtp_password = 'PASSWORD'

subject = 'INTRUDER ALERT!'
message = 'CALL POLICE ASAP, SOMEONE IS IN YOUR HOUSE!!!'

# Create a MIME message
msg = MIMEText(message)
msg['Subject'] = subject
msg['From'] = sender_email
msg['To'] = recipient_email

try:
    # Connect to the SMTP server
    server = smtplib.SMTP(smtp_host, smtp_port)
    server.starttls()
    server.login(smtp_username, smtp_password)

    # Send the email
    server.send_message(msg)
    print('Email sent successfully')
except smtplib.SMTPException as e:
    print(f'Failed to send email: {e}')
finally:
    # Disconnect from the SMTP server
    server.quit()

def run_server():
    host = '192.168.100.43'
    port = 8080

    # Create a socket object
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Allow socket reuse to avoid "Address already in use" error
    server_socket.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)

    # Bind the socket to a specific host and port
    server_socket.bind((host, port))

    # Listen for incoming connections
    server_socket.listen(5) # Allow up to 5 pending connections
    print(f"Server listening on {host}:{port}")

    while True:
        # Accept a client connection
        client_socket, client_address = server_socket.accept()

        # Handle the client (Arduino) connection
```



```
        handle_client(client_socket)

    # Continue accepting connections from the same Arduino

# Close the server socket
server_socket.close()
print("Server stopped")

if __name__ == '__main__':
    run_server()
```

Rezultate Obținute

Concluzii

În trecut nu eram tocmai pasionat de Hardware, însă acest proiect m-a făcut să realizez că e misto și această parte a industriei. Mi-a plăcut super mult să mă interesez ce piese să cumpăr și ce vreau de la proiect.

→ Am renunțat pe parcurs la tastatură, deoarece nu aveam destui pini disponibili pe Arduino, astfel aș fi trebuit să folosesc 2 plăcuțe Arduino și cred că dificultatea la proiectul meu a venit din alta parte și nu voiam să mă complic.

→ Am renunțat în timpul realizării proiectului la LED-ul pe care voiam să-l pun. Nu aveam cum să-l fac vizibil prin capacul cutiei (neavând un breadboard micuț), așadar am renunțat la el.

→ Am avut multe probleme cu cititoarele de RFID, încât am cumpărat 3 (din același loc), și am primit chinezarii (toate 3) care merg doar uneori și nu cu tag-ul albastru ci doar cu cel alb. Asta a fost în mare dificultatea proiectului și mai ales panica de a nu ști când îți va merge cititorul sau nu...

Mi-a plăcut super mult partea de networking & hardware și aș vrea să mă interesez mai mult ulterior, neavând timpul necesar acum.

Download

Arhiva conține fișierul corespunzător serverului și fișierul corespunzător logicii de pe plăcuța Arduino. Nu am scris un README, deoarece am explicat destul de clar ce am construit. Pentru mai multe nelămuriri contact: mihai.orange7@gmail.com

Link arhiva: pm_prj2023_ca_mihai.soare.zip

Jurnal

- 8 mai - toate piesele au ajuns
- 15 mai - am lipit cititorul de carduri RFID
- 21 mai - am testat toate componentele si am pregatit bibliotecile necesare pt fiecare componenta
- 29-30 mai - ultima noapte de dragoste, intaia noapte de razboi, proiectul a fost terminat si documentatia s-a facut

Bibliografie/Resurse

Bibliotecile folosite sunt urmatoarele:

- NewPing → https://github.com/eliteio/Arduino_New_Ping
- SPI si MFRC522 → MFRC522 este luata din Library Manager, iar SPI este inclusa deja
- LiquidCrystal_I2C → <https://github.com/fdebrabander/Arduino-LiquidCrystal-I2C-library>
- SoftwareSerial → inclusa in Arduino AVR

Toate componentele sunt luate de pe cleste.ro. Buget: 300 RON

Cont pe SMPT2GO pentru folosirea unui server free de SMTP.

[Export to PDF](#)

From:

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

Permanent link:

<http://ocw.cs.pub.ro/courses/pm/prj2023/apredescu/sistem-alarma>



Last update: **2023/05/30 13:12**