Multifunctional fingerprint lock door

Introducere

- Pentru a deschide usa, utilizatorul va trebui sa puna degetul pe cititorul de amprenta si se va verifica daca amprenta corespunde cu cea din sistem.
- Exista posibilitatea sa schimbe amprenta prin introducerea unei noi amprente. Prea multe incercari esuate de introducere a amprentei va rezulta in pornirea alarmei.
- Exista si o sonerie care poate fi pornita si un senzor care indica daca o persoana se afla in fata usii.

Schema functionala

Modul de functionare:

- Cand se apasa butonul de sonerie (si se mentine apasat), buzzerul va canta o melodie (luata din exemplele de la laborator).
- Cand senzorul ultrasonic detecteaza miscare (intr-o zona in care nu ar trebui sa existe miscare) va suna alarma.
- Pentru a deschide usa trebuie mai intai sa apasam pe butonul de deschidere/inchidere, apoi senzorul de amprente va citi imaginea data.
- Daca senzorul de amprente are in sistem amprenta introdusa, usa se va deschide.
- Cand usa este deschisa, alarma este dezactivata.
- Pentru a inchide usa se apasa pe butonul deschidere/inchidere.

Hardware Design

Componente hardware necesare pentru design:

```
• Cititor de amprente
• Senzor
```

- Buzzer
- Placa Arduino
- Servomotor

Conectare fingerprint: Fingerprint sensors - Arduino

- VCC - 5V/3.3V
- TX - RX (digital 2, software serial)
- RX - TX (digital 3, software serial)
- GND - GND

Software Design

Cod:

Ultrasonic ultrasonic(A2, 12); *A2- trigger, 12- echo Servo myservo;* create servo object to control a servo

SoftwareSerial mySerial(2, 3); *RX and TX for fingerprint Adafruit_Fingerprint finger = Adafruit_Fingerprint(&mySerial); int fingerprintID = 0; const int buttonPinAlarm = A1;* Button connected to analog pin A1; ring button

const int buttonPinPass = A0; *Button connected to analog pin A1; open/close door button const int ledPin = 5;* LED connected to digital pin 5

int previousButtonState, presentButtonState, ledState;

*the pin for buzzer int buzzer = 11; </note> Sonerie : <note tip> void buzz_function(){ for (int thisNote = 0; thisNote < notes * 2; thisNote = thisNote + 2) {* stop pressing the button

```
  if(digitalRead(buttonPinAlarm) == LOW){
    return;
  }
```

```
  // calculates the duration of each note
  divider = pgm_read_word_near(melody+thisNote + 1);
  if (divider > 0) {
    // regular note, just proceed
    noteDuration = (wholenote) / divider;
  } else if (divider < 0) {
    // dotted notes are represented with negative durations!!
    noteDuration = (wholenote) / abs(divider);
    noteDuration *= 1.5; // increases the duration in half for dotted notes
```

```
  }
```

```
  // we only play the note for 90% of the duration, leaving 10% as a pause
  tone(buzzer, pgm_read_word_near(melody+thisNote), noteDuration * 0.9);
```

```
  // Wait for the specief duration before playing the next note.
  delay(noteDuration);
```

```
  // stop the waveform generation before the next note.
  noTone(buzzer);
}
```

```
}
```

void setup() {

```
myservo.attach(6);  // attaches the servo on pin 9 to the servo object
Serial.begin(9600);
finger.begin(57600);
```

```
pinMode(buttonPinAlarm, INPUT);  // Set button pin as input
pinMode(ledPin, OUTPUT); // ledPin output
pinMode(buttonPinPass, INPUT);
previousButtonState = 0;
presentButtonState = 0;
ledState = LOW;
delay(50);
```

```
while (!finger.verifyPassword()) {
  Serial.println("Did not find fingerprint sensor :(");
  delay(300);
```

} Serial.println(“Found fingerprint sensor!”);

myservo.write(180);

}

Main function

void loop() {

```
previousButtonState    = presentButtonState;
presentButtonState = digitalRead(buttonPinPass);
```

```
while(ultrasonic.read() < 50 && !ledState){
  tone(buzzer, NOTE_D8, 10);  // alarm on
  delay(50);
```

```
}
```

```
if(previousButtonState == HIGH && presentButtonState == LOW) {/* if the
previous state is the HIGH and present state is LOW then  */
  int found = 1;
```

```
  while(finger.getImage() != FINGERPRINT_OK && ledState != HIGH){
    previousButtonState    = presentButtonState;
    presentButtonState = digitalRead(buttonPinPass);
    if(previousButtonState == HIGH && presentButtonState == LOW) {
      found = 0;
      break;
    }
    delay(100);
```

```
  }

  if(found){
     // found a match!
    Serial.print("Found ID #");
    Serial.print(finger.fingerID);
```

```
    ledState = !ledState; // change the state of the LED
```

```
    digitalWrite(ledPin, ledState); /* write that changed state to the LED
*/
```

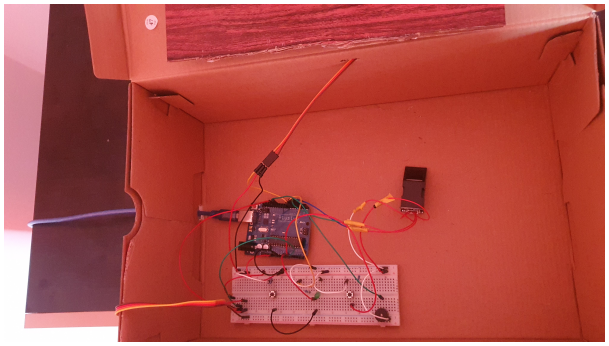
```
    // move the servo
    if(ledState){
      // open door
      for (int pos = 180; pos >= 90; pos -= 1) {
        myservo.write(pos);
        delay(15);
      }
    }
    else{
        // close door
        for (int pos = 90; pos <= 180; pos += 1) {
          myservo.write(pos);
          delay(15);
        }
      }
    }
```

```
}
```

```
// play song
if(digitalRead(buttonPinAlarm) == HIGH){
  buzz_function();
}
```

}

Concluzii



Pentru realizarea proiectului am folosit urmatoarele biblioteci:

#include <Adafruit_Fingerprint.h>

#include <Servo.h>

#include <Ultrasonic.h>

Am conectat senzorul fingerprint la pinii care permit comunicarea cu placa Arduino. Servo motorul deschide si inchide usa de carton.

Download

https://github.com/GabrielAndrei17/Proiect-PM

Jurnal

- 03.05.2023 - Realzarea Documentatiei
- 17.05.2023 - Realizarea proiect hardware.
- 24.05.2023 - Realizarea proiect software.

- 28.05.2023 - Update Documentatie.

Bibliografie/Resurse

Listă cu documente, datasheet-uri, resurse Internet folosite:

- https://github.com/adafruit/Adafruit-Fingerprint-Sensor-Library
- Laborator 1
- Laborator 2
- Laborator 3