2FA Vault

Introducere

Seiful 2FA reprezinta un mod inteligent de stocare a lucrurilor valoroase. Codul pin pentru deschiderea seifului nu este un cod fix, prestabilit, ci este generat si schimbat la cateva secunde de o aplicatie, similar logarii de tip “Two Factor Authentication”. Daca codul introdus este corect, utilizatorul va putea apropia mana de senzor si usa se va deschide singura.

Scopul acestui proiect este de a imbunatati modelul clasic de seif cu un pin de 4 cifre si de a oferi o securitate mult mai mare. Multi utilizatori seteaza coduri pin simple si usor de retinut, de exemplu “0000”, “1234” sau anul nasterii, care pot fi foarte usor ghicite de alte persoane. Prin folosirea acestei metode, codul va fi aproape imposibil de ghicit, iar utilizatorul nu va mai purta grija memorarii unui cod, intrucat il poate vedea pe telefon.

Am pornit de la ideea unui seif normal, observand tendinta oamenilor de a seta coduri pin cat mai simpliste, si m-am gandit sa aduc o imbunatatire in domeniul securitatii.

Descriere generală

Schema bloc:

Proiectul are la baza o placuta Arduino UNO R3 si foloseste urmatoarele componente:

- **Servomotor MG996**: deschide/inchide usa seifului
- **Keypad**: folosit la introducerea codului
- **Buzzer**: scoate un sunet de succes/eroare daca codul introdus este corect sau nu
- **Senzor de distanta cu infrarosu**: detecteaza apropierea mainii pentru a activa deschiderea usii
- **Ecran LCD 16×2**: vizualizarea cifrelor introduse
- **RTC module DS3231**: folosit pentru algoritmul de generare al codului pe placuta
- **4 baterii AA**: alimenteaza intreg sistemul

Placuta va calcula pe baza algoritmului TOTP un cod temporar, fiind acelasi algoritm folosit de “Google Authenticator”. Avand aceeasi cheie secreta, ambele coduri generate vor fi identice, astfel ca nu este nevoie de o comunicatie intre telefon si placuta.

Hardware Design

Componente folosite:

- Arduino UNO R3
- Servomotor MG996
- Miniservo SG90
- Senzor distanta cu infrarosu
- Buzzer pasiv
- Keypad 4×3
- Ecran LCD 16×2
- Modul RTC DS3231
- Suport baterii AA
- 8 baterii AA (1.5V)
- LED rosu
- LED verde

Schema poate fi analizata mai in detaliu dand click aici.

Schema electrica: schema_electrica_claudiu.pdf

BOM:

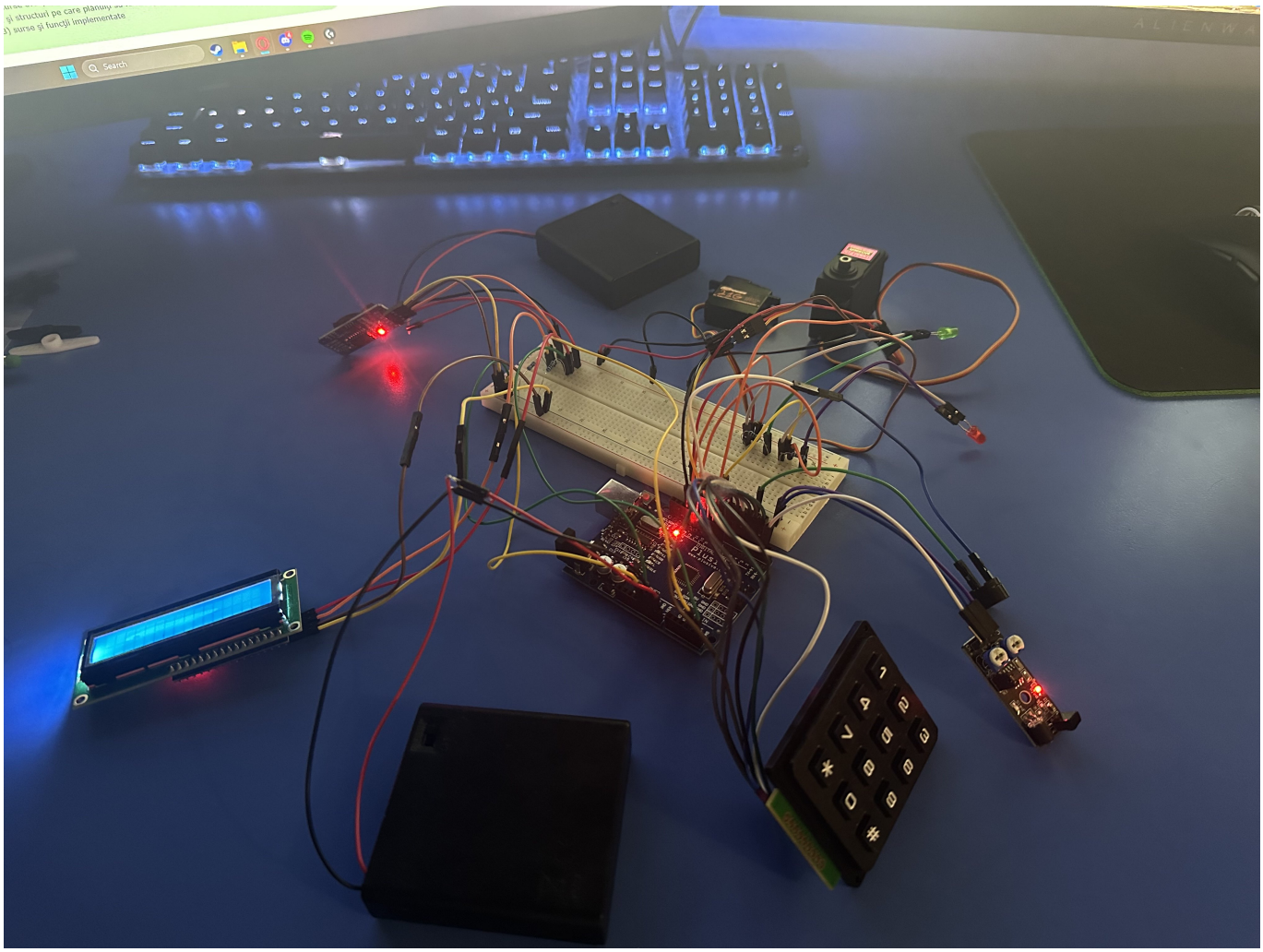
Componenta	Cantitate	Locul achizitionarii	Datasheet
Arduino Uno R3	1	Arduino	Datasheet
LED rosu	1	LED rosu	
LED verde	1	LED verde	
Rezistenta 220Ω	2	Rezistenta 220	
Rezistenta 10KΩ	2	Rezistenta 10k	
Buzzer pasiv	1	Buzzer	
LCD 16×2 (I2C)	1	Ecran LCD	
Keypad 4×3	1	Keypad	
Baterie AA	8	Baterii	
Suport baterii	2	Suport baterii	
RTC DS3231	1	Modul RTC	
Senzor distanta IR	1	Senzor distanta	
Servomotor MG996	1	Servomotor MG996	
Servomotor SG90	1	Servomotor SG90	

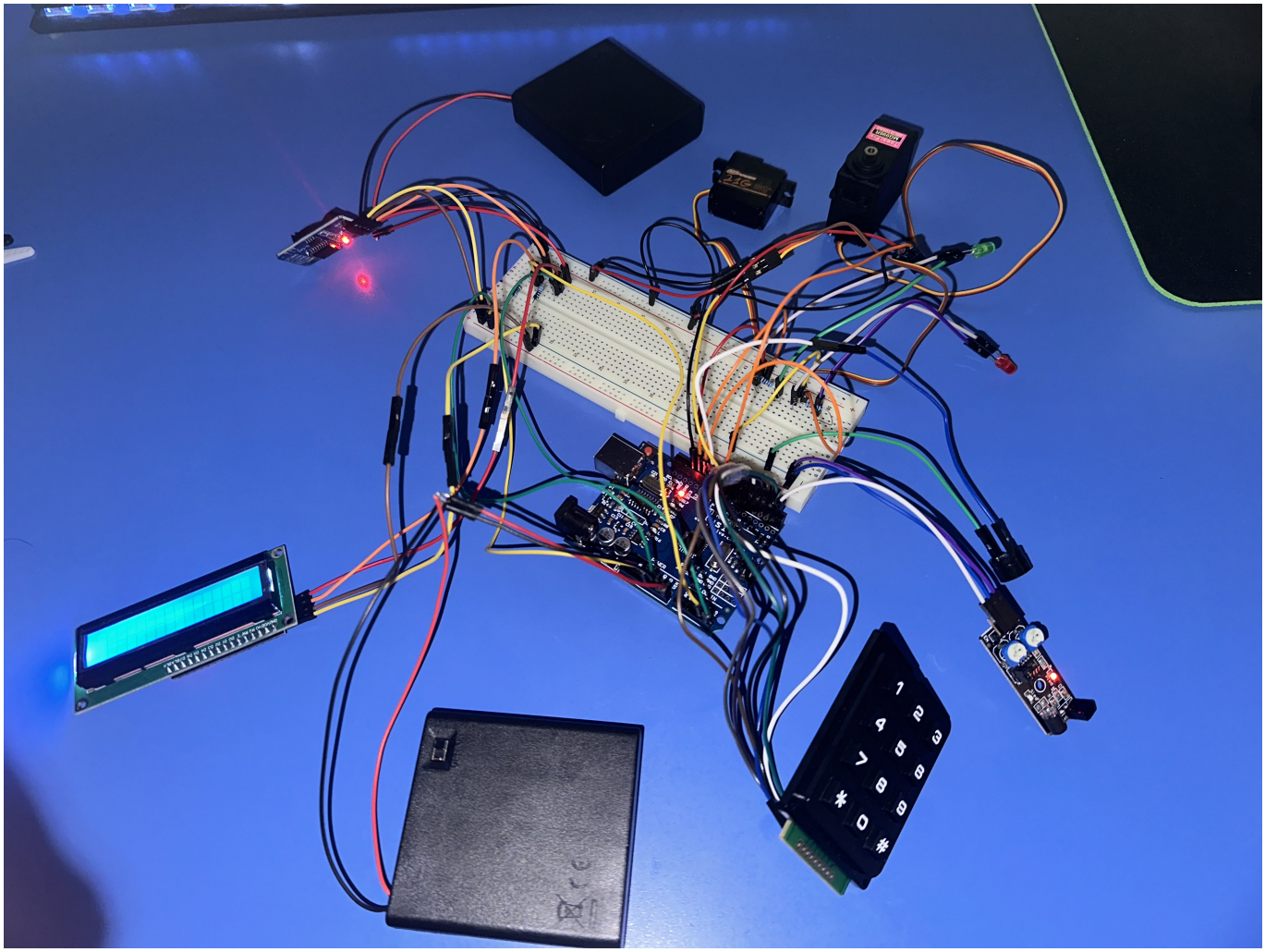
Descrierea componentelor:

- **Keypad 4×3** (D2 - D8): se foloseste pentru a introduce codul
- **LCD 16×2** (A4, A5): afiseaza cifrele introduse
- **RTC module DS3231** (A4, A5): mentine ora si data actuala pentru algoritmul TOTP de generare a codului
- **Baterii si suport** (VIN, GND): un set de baterii alimenteaza placuta Arduino, iar celalalt cele 2 servomotoare
- **Servomotor MG996** (D9): deschide/inchide usa seifului
- **Servomotor SG90** (D10): actioneaza incuietoarea seifului
- **Senzor distanta infrarosu** (D1): detecteaza apropierea mainii pentru a activa mecanismul de deschidere

- **LED-uri** (D12, D13): LED-ul rosu este aprins atunci cand seiful este incuiat, iar cel verde se aprinde cand seiful este descuiat
- **Buzzer pasiv** (D11): face un sunet de succes/esec atunci cand este introdus un cod

Se foloseste ground comun intre alimentarea placutei si alimentarea celor doua motorase.





Software Design

Descrierea codului aplicaţiei (firmware):

- mediu de dezvoltare (if any) (e.g. AVR Studio, CodeVisionAVR)
- librării şi surse 3rd-party (e.g. Procyon AVRlib)
- algoritmi şi structuri pe care plănuiţi să le implementaţi
- (etapa 3) surse şi funcţii implementate

Mediu de dezvoltare: PlatformIO

Biblioteci:

- Arduino.h - functionalitati de baza Arduino
- Keypad.h - ajuta cu citirea de la tastatura si ajuta la claritate in cod (sub forma de matrice)
- LiquidCrystal_I2C.h - ajuta la afisarea pe ecranul LCD
- RTClib.h - pentru functionarea modulului RTC; retine ora si data exacta in cod
- TOTP.h - genereaza codul temporar
- Base32.h - decodeaza cheia secreta in baza 32

Element de noutate: Deblocarea seifului se face prin autentificarea in 2 pasi; codul pentru seif se schimba la fiecare 30 de secunde si poate fi gasit pe aplicatia Google Authenticator dupa scanarea

unui cod QR

Functionalitati din laboratoare: GPIO, USART, PWM, I2C

Algoritm:

- in functia **setup()** se initializeaza toti pinii si variabilele necesare
- la fiecare iteratie din functia **loop()** se genereaza codul temporar folosind data si ora exacte, retinute intr-o variabila de tip RTC_DS3231
- se primeste un simbol de la tastatura
- daca simbolul este '*' se aplica logica de backspace - se sterge ultimul caracter din sirul introdus si este reafisat noul cod
- daca simbolul este '#' se aplica logica de introducere cod astfel:
  - daca seiful este blocat se verifica daca codul introdus este corect sau nu; in caz afirmativ seiful trece in starea de “deblocat”
  - daca seiful este deblocat atunci seiful se va inchide si bloca automat si va trece in starea de “blocat”
- daca simbolul este o cifra aceasta va fi atasata la sirul deja introdus, cu o singura conditie: se pot introduce maxim 6 cifre (aceasta este lungimea codului)
- daca seiful este **deblocat** si senzorul de distanta detecteaza un obstacol usa se va deschide automat

Codul este creat individual pe placuta si verifica daca codul introdus este egal cu codul generat, astfel ca aplicatia si placuta lucreaza independent una de alta. Ambele folosesc algoritmul TOTP cu aceeasi cheie secreta si se poate garanta ca vor genera acelasi cod. Modulul RTC retine data si ora exacta, astfel se asigura faptul ca cele doua coduri sunt sincronizate (daca modulul ar retine ora cu 3 secunde in urma, atunci codul va fi actualizat cu 3 secunde intarziere, de unde rezulta importanta sincronizarii la secunda).

Repartizare pini:

```
#define IR_SENSOR_PIN PD0      // Arduino D0
#define MG996_SERVO_PIN PB1     // Arduino D9 (OCR1A)
#define GS21G_SERVO_PIN PB2    // Arduino D10 (OCR1B)
#define BUZZER_PIN PB3         // Arduino D11
#define BUZZER 11
#define RED_LED PB4            // Arduino D12
#define GREEN_LED PB5          // Arduino D13
```

[GPIO] Ledurile, buzzerul si senzorul infrarosu au fost declarate si setate astfel:

```
// Set LEDs, buzzer and servos as output
DDRB |= (1 << RED_LED) | (1 << GREEN_LED) | (1 << BUZZER_PIN)
                | (1 << MG996_SERVO_PIN) | (1 << GS21G_SERVO_PIN);

// Set IR sensor as input with pull-up
DDRD &= ~(1 << IR_SENSOR_PIN);      // Set as input
PORTD |= (1 << IR_SENSOR_PIN);      // Enable pull-up resistor

start_led(RED_LED); // acest apel executa aceasta linie PORTB |= (1 << PB4);
```

Tastatura a fost initializata folosind libraria **Keypad.h** si este retinuta ca matrice de 4×3.

[PWM] Motorasele au fost setate astfel:

```
// Set PWM pins as outputs
pinMode(MG996_SERVO_PIN, OUTPUT);
pinMode(GS21G_SERVO_PIN, OUTPUT);

// Configure Timer1 for Fast PWM, 20ms period (50Hz)
TCCR1A = (1 << COM1A1) | (1 << COM1B1) | (1 << WGM11); // Non-inverting PWM
A & B
TCCR1B = (1 << WGM13) | (1 << WGM12) | (1 << CS11);    // Prescaler 8
ICR1 = 39999; // TOP = 20ms at 16MHz/8 => 0.5 µs per tick

DDRB |= (1 << PB1) | (1 << PB2);
```

[I2C] Ecranul LCD si modulul RTC au fost initializate folosind librarii externe:

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

RTC_DS3231 rtc;
LiquidCrystal_I2C lcd(0x27, 16, 2);

lcd.init();
lcd.backlight();
lcd.clear();
lcd.setCursor(0, 0);

if (!rtc.begin()) {
        Serial.println("RTC not found!");
        while (1);
}
```

[USART] Serialul este folosit pentru debug si este initializat astfel:

```
Serial.begin(9600);
```

Serialul este folosit in principal pentru debug si afisarea erorilor.

Pentru a asigura ca motorasele se misca lin si la o viteza rezonabila a fost folosita urmatoarea functie pentru a schimba gradual unghiul la care sa se miste motorasul:

```
void smoothServoMove(uint8_t pin, int fromAngle, int toAngle, int stepDelay
= 15) {
        if (fromAngle == toAngle) {
                setServoAngle(pin, toAngle);
                return;
        }

        int stepDirection = (toAngle > fromAngle) ? 1 : -1;
```

```
        for (int angle = fromAngle; angle != toAngle; angle += stepDirection)
{
                setServoAngle(pin, angle);
                delay(stepDelay);
        }

        // Ensure the final angle is set precisely
        setServoAngle(pin, toAngle);
}
```

Calibrari:

- senzorul de distanta este setat sa detecteze la o distanta mica pentru a evita situatia ca senzorul sa detecteze ceva prea indepartat si sa deschida usa fara a dori acest lucru
- modulul RTC trebuie calibrat cu data si ora exacte dupa ce acesta pierde alimentarea de la curent (exista functia ajutatoare setRTC pentru a facilita acest lucru)
- a fost introdus un delay intre activarea motoraselor pentru a nu se suprapune actiunea acestora (ex: al doilea motor sa deschida usa inainte ca primul motor sa inlature incuietoarea)
- la pornirea placutei incuietoarea este ridicata, usa adusa in pozitia de “inchis” si incuietoarea este lasata jos pentru a asigura faptul ca seiful porneste ca fiind incuiat

Rezultate Obţinute

Ca rezultat am obtinut un seif cu deschidere/inchidere automata si cu un nivel de securitate marit datorita codului temporar (acesta fiind aproape imposibil de ghicit - probabilitate 1 la un milion). In acelasi timp, acesta este foarte usor de folosit, deoarece codul se afla permanent pe telefon si poate fi adaugatfoarte usor, doar scanand codul QR.

Concluzii

Proiectul a indeplinit asteptarile initiale si se comporta cum ar trebui. Implementarea software a fost relativ simpla, partea dificila fiind conectarea tuturor componentelor si functionarea simultana a acestora. Totodata proiectul a aratat ca se poate face ceva util si interesant folosind o placuta Arduino si niste componente.

Download

O arhivă (sau mai multe dacă este cazul) cu fişierele obţinute în urma realizării proiectului: surse, scheme, etc. Un fişier README, un ChangeLog, un script de compilare şi copiere automată pe uC crează întotdeauna o impresie bună 😉.

Fişierele se încarcă pe wiki folosind facilitatea **Add Images or other files**. Namespace-ul în care se încarcă fişierele este de tipul **:pm:prj20??:c?** sau **:pm:prj20??:c?:nume_student** (dacă este cazul). **Exemplu:** Dumitru Alin, 331CC → **:pm:prj2009:cc:dumitru_alin**.

Jurnal

Puteți avea și o secțiune de jurnal în care să poată urmări asistentul de proiect progresul proiectului.

Bibliografie/Resurse

Github

Export to PDF