

Sistem antifurt

Proiect Sisteme Incorporate

Colaboratori: Paulescu Andrei
Raduțu Robert-Cristian
Rîcu Alexandru
Stoica Andra-Cristiana

Cuprins

Introducere	4
Descriere Componente	5
<i>Arduino UNO</i>	<i>5</i>
<i>Microtastatură/keypad</i>	<i>9</i>
<i>Modul Rfid</i>	<i>12</i>
<i>Cameră ESP32</i>	<i>16</i>
Scenariu Funcționare.Diagrama UML	20
Schema circuitului	21
Interconectarea componentelor	22
Cod sursă	23
Referințe / Bibliografie.....	26

Introducere

Titlul lucrării: Sistem Antifurt

Tema lucrării:

Detectarea automată a persoanelor și deschiderea ușii în cazul confirmării identității acestora.

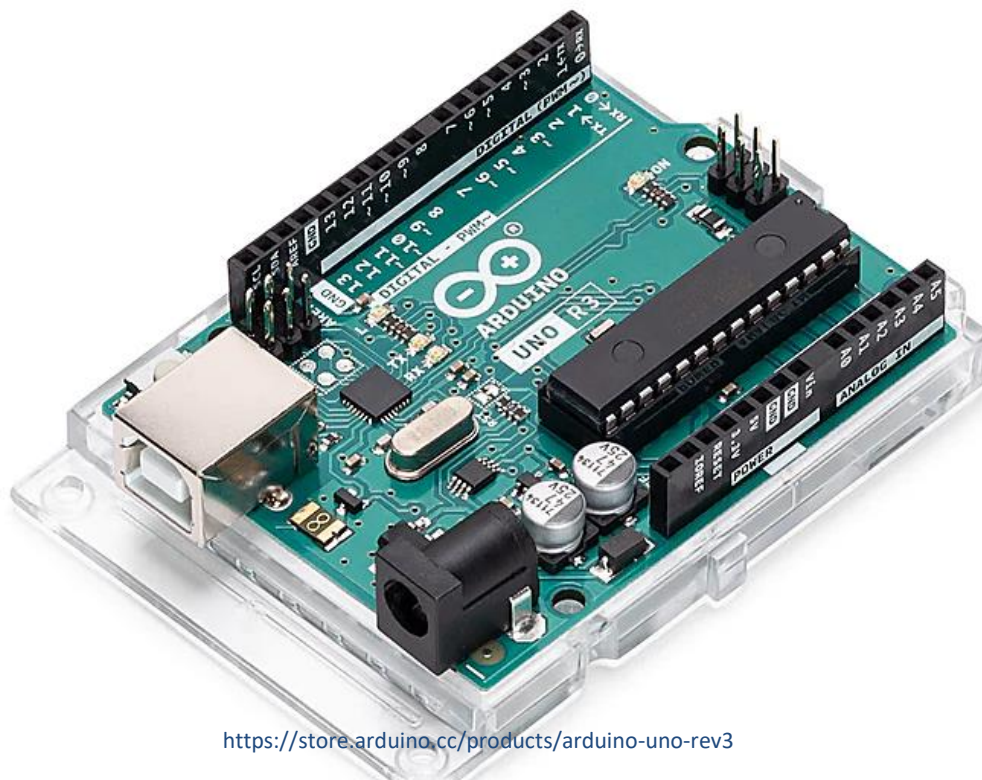
Cerințe:

- Se vor utiliza module (rfid, camera, senzor de lumină) pentru detectarea unei persoane, respectiv detectarea cardurile de acces scanate.
- Se va folosi o cameră pentru a captura persoanele care doresc să acceseze încăperea.
- Modulele vor comunica datele folosind module de comunicare (WiFi, interfețe seriale).
- Datele vor fi transmise spre un server web pentru procesare și analiza imaginilor.
- Parametrii mășurați (recunoaștere facială, fețe detectate) se vor afișa pe un server web.
- Datele preluate trebuie analizate, ușa trebuie descuiată/încuiată corespunzător.

Materiale necesare:

- Placă de dezvoltare Arduino Uno
- Modul camera ESP32
- Modul wifi ESP8266
- Modul rfid + carduri NFC
- Microtastatură 4*4
- Actuator închidere ușă
- Scriitor USB - TTL
- Releu 5VDC
- Conectori și socket-uri
- Fotorezistor
- Leduri, rezistențe
- Buzzer activ

Arduino Uno:



<https://store.arduino.cc/products/arduino-uno-rev3>

[1]

Arduino Uno este o placă de microcontroler bazată pe ATmega328P.

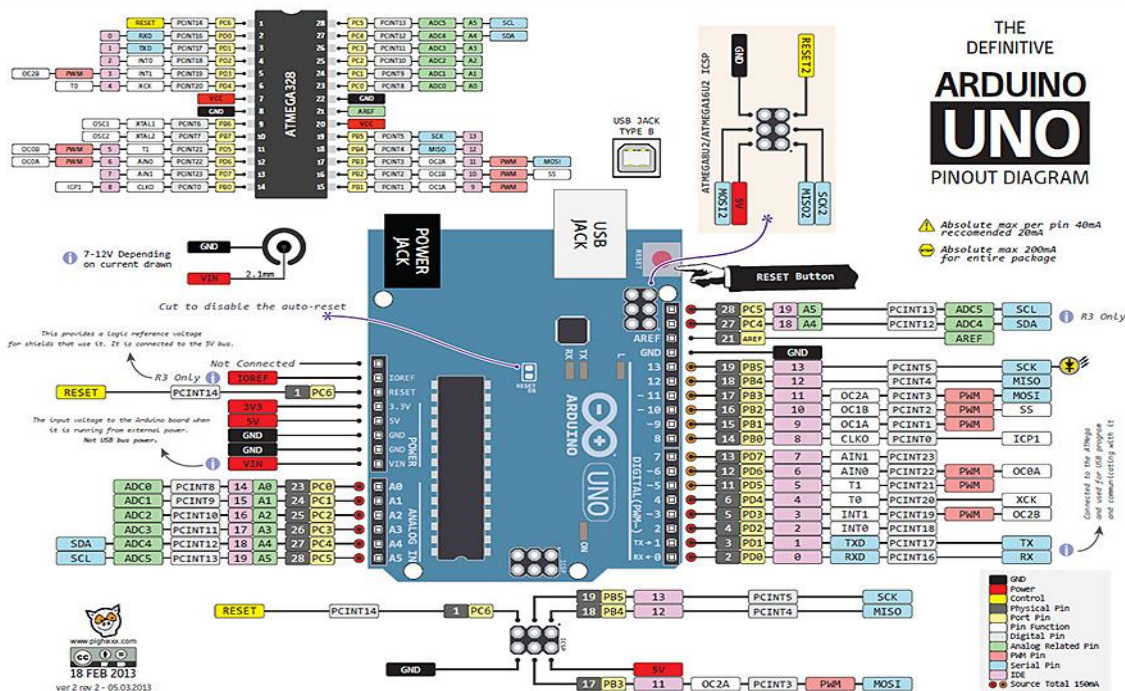
Are 14 pini digitali de intrare/ieșire (dintre care 6 pot fi utilizați ca ieșiri PWM), 6 intrări analogice, un rezonator ceramic de 16 MHz (CSTCE16M0V53-R0), o conexiune USB, o mufă de alimentare, 32KB Flash, 2KB SRAM, 1KB EEPROM, un antet ICSP și un buton de resetare.

Aceasta conține tot ceea ce este necesar pentru a sprijini microcontrolerul, pur și simplu se conectează la PC printr-un cablu USB pentru scrierea programului și pentru alimentare sau poate fi alimentat cu un adaptor AC-to-DC sau de la o baterie.

Aceasta este încă utilizată pe scară largă pentru diverse scopuri educaționale și proiecte științifice. Standardul ridicat al plăcii și performanța de bună calitate fac din aceasta o resursă excelentă pentru a capta în timp real de la senzori și pentru a utiliza diferite periferice.

Pinout:

[5]



Analog la Digital Converter sau ADC este un circuit electronic utilizat pentru a converti semnalele analogice în semnale digitale. Această reprezentare digitală a semnalelor analogice permite procesorului (care este un dispozitiv digital) să măsoare semnalul analogic și să-l folosească în programul său.

Pini Arduino A0-A5 sunt capabili de a citi tensiuni analogice. Pe Arduino ADC-ul are rezoluție pe 10 biți, ceea ce înseamnă că poate reprezenta tensiunea analogică cu 1024 niveluri digitale. ADC convertește tensiunea în biți pe care microprocesorul îi poate înțelege. Pini 0-13 din Arduino Uno servesc ca pini digitali de intrare/ieșire, iar pinul 13 din Arduino Uno este conectat la LED-ul BUILT-IN.

Pin	Function	Type	Description
1	NC	NC	Not connected
2	IOREF	IOREF	Reference for digital logic V - connected to 5V
3	Reset	Reset	Reset
4	+3V3	Power	+3V3 Power Rail
5	+5V	Power	+5V Power Rail
6	GND	Power	Ground
7	GND	Power	Ground
8	VIN	Power	Voltage Input
9	A0	Analog/GPIO	Analog input 0 /GPIO
10	A1	Analog/GPIO	Analog input 1 /GPIO
11	A2	Analog/GPIO	Analog input 2 /GPIO
12	A3	Analog/GPIO	Analog input 3 /GPIO
13	A4/SDA	Analog input/I2C	Analog input 4/I2C Data line
14	A5/SCL	Analog input/I2C	Analog input 5/I2C Clock line

Pin	Function	Type	Description
1	D0	Digital/GPIO	Digital pin 0/GPIO
2	D1	Digital/GPIO	Digital pin 1/GPIO
3	D2	Digital/GPIO	Digital pin 2/GPIO
4	D3	Digital/GPIO	Digital pin 3/GPIO
5	D4	Digital/GPIO	Digital pin 4/GPIO
6	D5	Digital/GPIO	Digital pin 5/GPIO
7	D6	Digital/GPIO	Digital pin 6/GPIO
8	D7	Digital/GPIO	Digital pin 7/GPIO
9	D8	Digital/GPIO	Digital pin 8/GPIO
10	D9	Digital/GPIO	Digital pin 9/GPIO
11	SS	Digital	SPI Chip Select
12	MOSI	Digital	SPI1 Main Out Secondary In
13	MISO	Digital	SPI Main In Secondary Out
14	SCK	Digital	SPI serial clock output
15	GND	Power	Ground
16	AREF	Digital	Analog reference voltage
17	A4/SDA	Digital	Analog input 4/I2C Data line (duplicated)
18	A5/SD5	Digital	Analog input 5/I2C Clock line (duplicated)

În cazul plăcii Arduino Uno pinii 3,5,6,9,10,11 au capacitatea de a fi folosiți ca PWM. Pini I/O pot oferi până la 40 mA max, dar curentul recomandat este de 20 mA. Curentul maxim absolut furnizat de la toți pinii împreună este de 200mA.

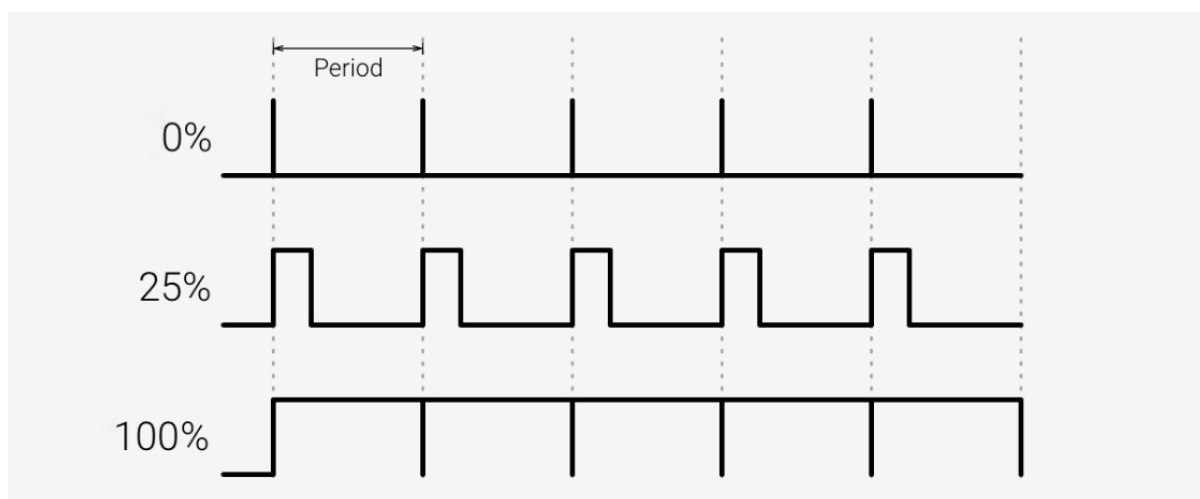
[7] Ce este PWM?

În general, Modulația lății pulsului (PWM) este o tehnică de modulare utilizată pentru a codifica un mesaj într-un semnal pulsatoriu.

Un PWM este compus din două componente cheie: frecvența și ciclul de funcționare.

- Frecvența PWM dictează cât timp este nevoie pentru a finaliza un singur ciclu (perioadă) și cât de repede fluctuează semnalul de la high la low.
- Ciclul de funcționare determină cât timp un semnal rămâne ridicat(high) din perioada totală. Ciclul de funcționare este reprezentat în procente.

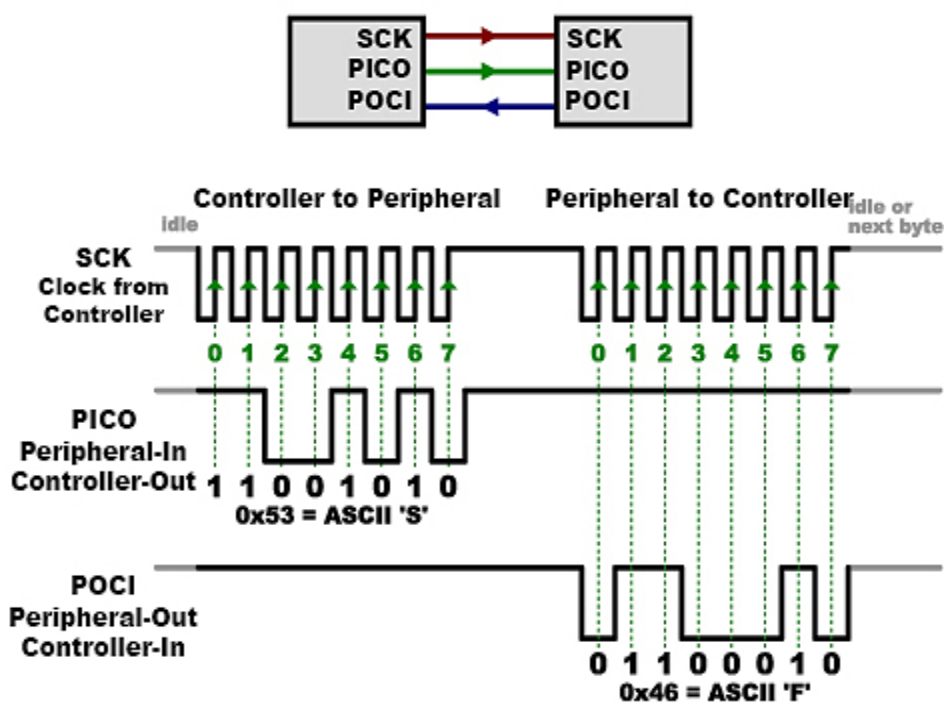
În Arduino, pinii de PWM produc o frecvență constantă de $\sim 500\text{Hz}$, în timp ce ciclul de funcționare se modifică în funcție de parametrii stabiliți de utilizator. Vedeți următoarea ilustrație:



[6]

Ce este SPI?

Serial Peripheral Interface (SPI) este un protocol de transmisie de date serial utilizat de microcontrolere pentru a comunica cu unul sau mai multe dispozitive externe într-o conexiune de tip magistrală. SPI poate fi, de asemenea, utilizat pentru a conecta 2 microcontrolere. Pe magistrala SPI, există întotdeauna un dispozitiv care este notat ca un dispozitiv master(microcontroler) și toate celelalte ca slaves(periferice).



<https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi>

Functionare:

- Pinul SS/CS (Slave Select/Circuit Select) determină dispozitivul cu care masterul comunică în prezent.
- POCI / MISO (Master In Slave Out) - O linie pentru trimiterea de date de la periferic către dispozitivul Master.
- PICO / MOSI (Master Out Slave In) - Linia principală pentru trimiterea de date de la Master către dispozitivele periferice.
- SCK (Serial Clock) - Un semnal de ceas/tact generat de dispozitivul Master pentru a sincroniza transmisia de date.

4x4 Keypad

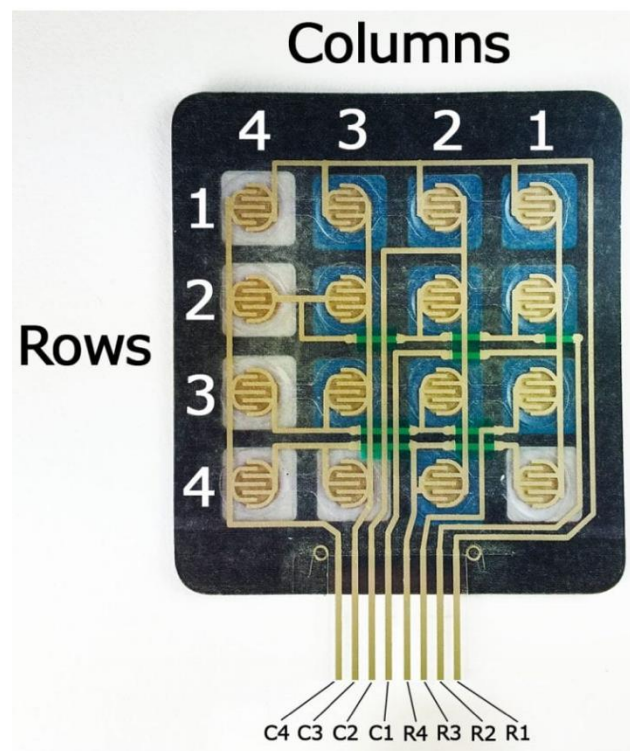
[2]

Keypad-urile reprezintă o metodă bună de a lăsa user-ul să interacționeze cu proiectul nostru. Pot fi folosite pentru a naviga prin meniuri, a introduce parole, a controla jocuri sau roboți.

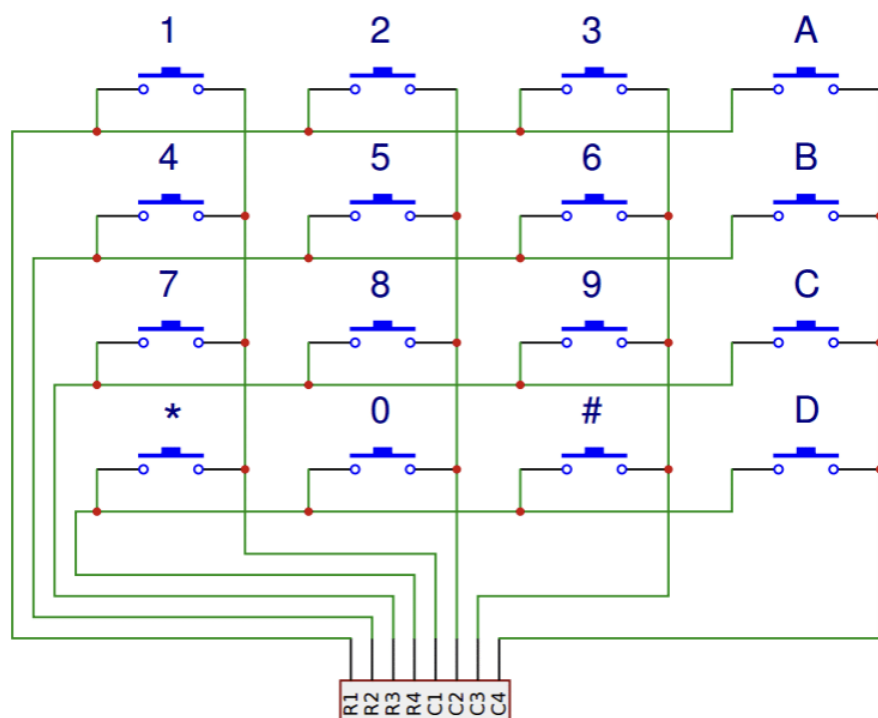
Prima dată să înțelegem cum Arduino detectează apăsările de taste.



Sub fiecare tastă este un întrerupător din membrană. Fiecare întrerupător dintr-o linie este conectat către celelalte întrerupătoare din aceea linie printr-o urma conductivă. Prin apăsarea unui buton se închide întrerupătorul dintre o coloană și o linie, permitând curentului să treacă între un pin al coloanei și un pin al liniei.



Schema unui 4x4 KeyPad, arată cum liniile si coloanele sunt conectate.

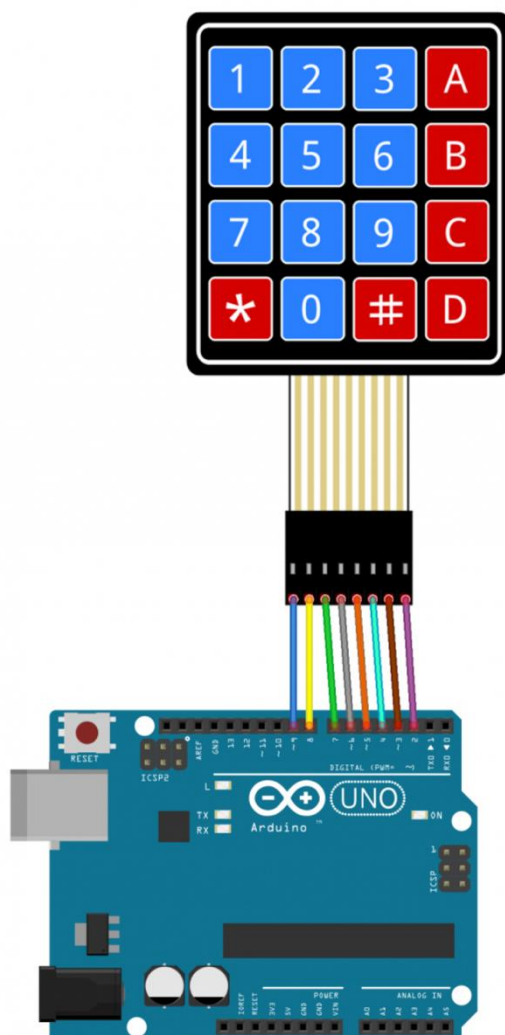


Arduino detectează ce buton este apăsat detectând pinul de rând și coloană care este conectat la buton.

Acest lucru se întâmplă în patru pași:

1. În primul rând, când nu este apăsat niciun buton, toți pinii coloanei sunt ținuti pe HIGH, iar toți pinii sunt ținuti LOW;
2. Când este apăsat un buton, pinul coloanei este tras la nivel LOW, deoarece curentul de la coloana cu nivel HIGH curge spre pinul rândului cu nivel LOW;
3. Arduino știe acum în ce coloană se află butonul, deci trebuie doar să găsească rândul în care se află butonul. Face acest lucru prin comutarea fiecăruia dintre pinii rândului la nivel HIGH, și în același timp citind toți pinii coloanei pentru a detecta care pin de coloană revine la nivel HIGH;
4. Când pinul coloanei devine din nou HIGH, Arduino a găsit pinul rândului care este conectat la buton.

Conectarea pinului la Arduino



INSERT CODE HERE

Modulul RFID RC522



[3] Modulul RFID RC522, bazat pe circuitul integrat MFRC522 de la NXP, este una dintre cele mai ieftine opțiuni RFID pe care le poți achiziționa online, cu prețuri sub patru dolari. De obicei, acesta vine cu o etichetă RFID de tip card și o etichetă sub formă de breloc cu o memorie de 1KB. Un sistem RFID sau de identificare prin radiofrecvență este format din două componente principale: o etichetă/card atașată obiectului care trebuie identificat și un cititor care citește eticheta/cardul.

Cipul răspunde prin trimiterea informațiilor stocate înapoi la cititor sub forma unui alt semnal radio. Aceasta este numită reflexie înapoi (backscatter). Cititorul detectează și interpretează această reflexie înapoi și trimite datele către un computer sau un microcontroler.

Una dintre cele mai importante aspecte ale modelului RC522 este că poate scrie pe etichete, ceea ce înseamnă că poți stoca orice mesaj în ele. Modulul cititor RC522 RFID este conceput pentru a crea un câmp electromagnetic de 13,56 MHz și pentru a comunica cu etichetele RFID (etichete conform standardului ISO 14443A).

Cititorul poate comunica cu un microcontroler printr-o interfață SPI cu 4 pini, având o viteză maximă de transfer de date de 10 Mbps. De asemenea, acesta suportă comunicarea prin protocoalele I2C și UART.

Modulul RC522 RFID poate fi programat pentru a genera o întrerupere (interrupt), permițând alertarea atunci când o etichetă se apropie de aceasta, în loc să existe flag-ul/sa fie pusă întrebarea constant de către modul "Există o etichetă în apropiere?".

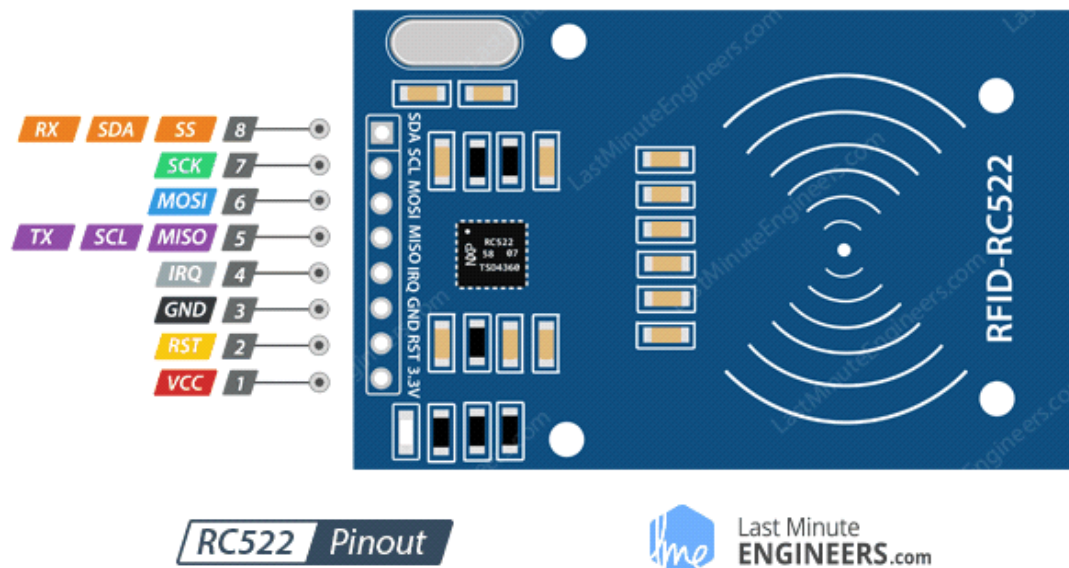
Tensiunea de alimentare a modulului variază între 2,5 și 3,3V, dar pinii logici suportă până la 5 volți, astfel că îl putem conecta ușor la un Arduino sau orice microcontroler cu logica de 5V fără a utiliza un convertor de nivel logic.

Specificații tehnice:

Frequency Range	13.56 MHz ISM Band
Host Interface	SPI / I2C / UART
Operating Supply Voltage	2.5 V to 3.3 V
Max. Operating Current	13-26mA
Min. Current(Power down)	10μA
Logic Inputs	5V Tolerant
Read Range	5 cm

RC522 RFID pini:








Modulul RC522 are în total 8 pini. Conexiunile sunt următoarele:



- Pinul VCC furnizează alimentarea modului. Aceasta poate fi în intervalul de la 2.5 la 3.3 volți. Poți conecta acest pin la ieșirea de 3.3V a Arduino-ului. Modulul poate suporta tensiuni până la 5V, dar aceasta tensiune de alimentare este nerecomandată.
- Pinul RST este o intrare pentru reset și oprire. Atunci când acest pin este setat la nivel scăzut (0), modulul intră în modul de oprire, în care oscilatorul este oprit și pinii de intrare sunt deconectați de la unitatea centrală. În sens opus, modulul este resetat pe frontul de înalt al semnalului (1).
- Pinul GND este pinul de masă și trebuie conectat la pinul GND al Arduino-ului.
- Pinul IRQ este folosit pentru întrerupere și alertează microcontrolerul atunci când o etichetă RFID se află în apropiere.

- Pinul MISO/SCL/Tx acționează ca master-in-slave-out (MISO) atunci când interfața SPI este activată, ca ceas serial (SCL) atunci când interfața I2C este activată sau ca ieșire de date serială (Tx) atunci când interfața UART este activată.
- Pinul MOSI acționează ca master-out-slave-in și este intrarea SPI către modulul RC522.
- Pinul SCK (Serial Clock) primește impulsurile de ceas furnizate de către bus-ul SPI master (Arduino).
- Pinul SS/SDA/Rx acționează ca intrare de semnal atunci când interfața SPI este activată, ca date seriale atunci când interfața I2C este activată și ca intrare de date seriale atunci când interfața UART este activată. Acest pin este în mod obișnuit marcat prin înconjurarea sa cu un pătrat, astfel încât poate fi folosit ca referință pentru identificarea altor pini.

Conectarea pinilor cu placa Arduino Uno:

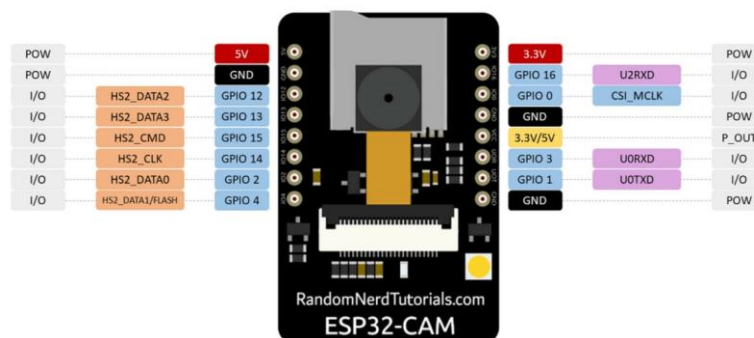
RC522 Module		Arduino
VCC		3.3V
GND		GND
RST		5
MISO / SCL / Tx		12
MOSI		11
SCK		13
SS / SDA / Rx		10

Esp32 – Cam

[4]

Esp32-Cam AI-Thinker pinout (Source: [Random Nerd Tutorials](#))

Esp32-Cam este un modul de cameră ieftin, cu 2 megapixeli, de dimensiuni reduse, dezvoltat de AI-Thinker. Integrează un Esp32, un singur SoC (System on a Chip) Wi-Fi și Bluetooth de 2.4 GHz proiectat de Espressif Systems și o cameră OV2640 de la OmniVision.



Modulul standard AI-Thinker

Esp32-Cam este compact (27

mm x 40.5 mm x 4.5 mm) și foarte ieftin (în jur de \$10). Chiar și versiunea cea mai scumpă a Esp32-Cam, modelul CH340, nu costă mai mult de \$20.

Esp32-Cam se mândrește cu o lanternă integrată și un slot pentru card microSD pentru a salva imaginile de la camera de 2 megapixeli. Același card microSD poate fi folosit pentru a stoca fișiere utile aplicației tale.

Împreună cu conectivitatea integrată Wi-Fi și Bluetooth (cu o antenă la bord), placa are un conector pentru o antenă externă, care poate fi utilă dacă trebuie să mărești raza semnalului tău.

Poți alimenta placa cu 3.3V sau 5V; alege doar pinul potrivit. Chiar dacă multe GPIO-uri sunt folosite de cameră (15 pini), adaptorul microSD (6 pini), lanterna integrată (GPIO 4) și LED-ul roșu integrat (GPIO 33), rămân 10 GPIO-uri care pot fi conectate la periferice externe (vezi etichetele albastre deschis în imaginea de mai sus).

Pe partea negativă, din cauza dimensiunilor sale convenabil de mici, nu există un programator USB integrat (doar Esp32-Cam CH340 îl are), iar comutatorul de resetare se află pe partea de jos, care nu este ușor accesibil.

Cu toate acestea, având una sau mai multe plăci Esp32-Cam la îndemână te poate ajuta să începi un proiect în care ai nevoie de o cameră mică și de o soluție ieftină "all-in-one".

ESP32-CAM WITH ARDUINO IDE: THE BASICS ARDUINO IDE



Esp32-Cam AI-Thinker (Source: Luca Dinale via All3DP)

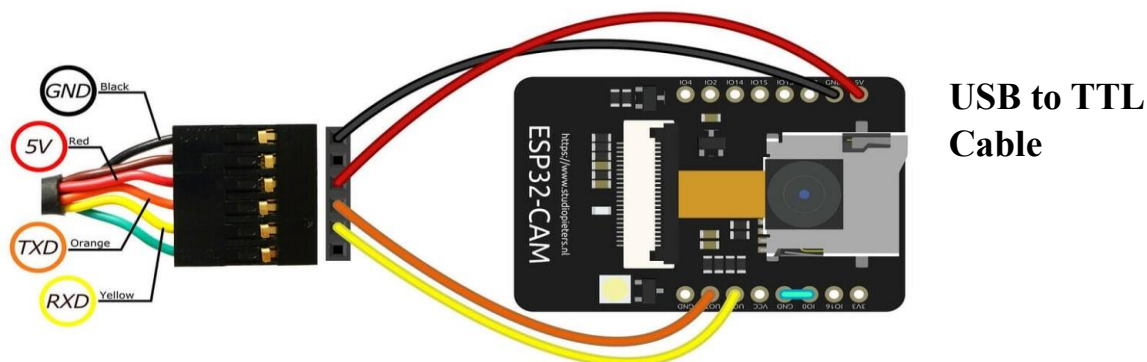
Cea mai simplă modalitate de a începe programarea acestei mici plăci este de a folosi lanțul de unelte Arduino Esp32 în Arduino IDE (Integrated Development Environment), o aplicație dezvoltată de Espressif care este folosită pentru a scrie și încărca coduri (programe) pe plăcile compatibile cu Arduino.

Arduino IDE folosește o versiune simplificată și redusă a limbajului de programare C++. Începătorii pot începe cu exemple și tutoriale, dar programatorii mai avansați pot crea propriile biblioteci și pot utiliza o aplicație CLI (Command Line Interface).

Cu Arduino, programul tău se numește "sketch", iar când îl compilezi, este "magic" depanat, tradus în C++ corespunzător și trimis plăcii tale în formă binară (limbajul mașinii).

Dacă întâmpini erori în timpul compilării, Arduino IDE îți dă sugestii pentru a-ți găsi greșeala. Destul de des, în special pentru utilizatorii noi, este o eroare de sintaxă. Una dintre cele mai frecvente erori este uitarea de a adăuga un punct și virgulă la sfârșitul unei linii. Poți afla mai multe despre Arduino IDE versiunea 1, Arduino IDE versiunea 2, Arduino Web Editor și Arduino PRO CLI de pe site-ul Arduino.

Există diferite modalități de a conecta Esp32-Cam și Arduino IDE (care rulează pe computerul tău), atât fizice, cât și la distanță. În continuare, vom discuta despre cinci modalități diferite de a face conexiunea fizică, apoi vom vedea cum să lucrezi cu Esp32-Cam și Arduino IDE, iar în final, vom examina alternativa: conectarea lucrurilor de la distanță.



Connection between an FTDI cable and an Esp32-Cam (Source: Luca Dinale via All3DP)

Cablu USB to TTL este un convertor de la USB la Serial care oferă o modalitate simplă de a conecta Esp32-Cam standard la un port USB de pe computerul tău. Dacă avem la îndemână un cablu USB to TTL, putem alimenta placa cu pinii de 3.3V sau 5V, plus pinii Ground (GND).

Pinul GPIO0 determină modul Normal sau modul Flash al modulului:

GPIO0 conectat la Ground = Esp32-CAM în modul Flash = Poți încărca codul

GPIO0 nu este conectat la Ground = Esp32-CAM în modul Normal = Placa execută programul

Există două pini seriali, GPIO01 și GPIO03 (marcați UOT și UOR), care pot fi utilizați pentru transmisie serială și pentru a primi date cu protocolul UART (Universal Asynchronous Receiver-Transmitter).

În afara cablului USB to TTL, vom avea nevoie de cinci cabluri jumper pentru a conecta Esp32-Cam, așa cum se vede în imaginea de mai sus:

Cablu roșu de 5V se conectează la 5V pe placă (sau 3.3V dacă folosești un cablu FTDI de 3.3V).

GND la GND.

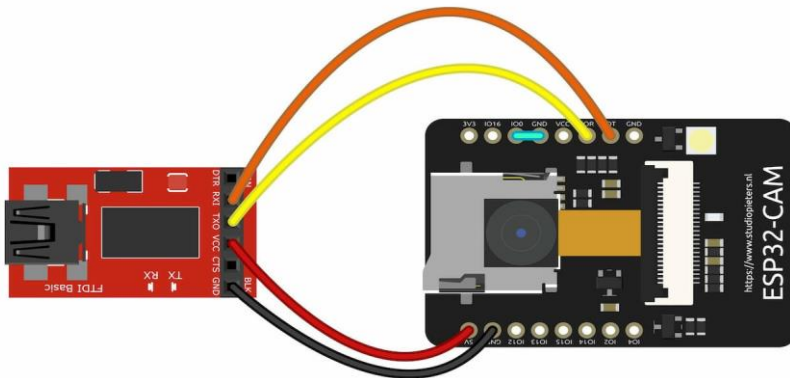
Cablu portocaliu TXD la UOT pe placă.

Cablu galben RXD la UOR pe placă.

Un jumper conectează GPIO0 și GND (înainte de încărcarea codului). Întotdeauna trebuie să ne asigurăm că resetăm placa după ce schimbăm starea GPIO0. Conectăm și resetăm pentru a încărca, deconectăm și resetăm pentru a rula programul.

De reținut este că, deoarece comutatorul de resetare este pe partea de jos a plăcii, poate fi greu de atins. Pentru a reseta placa, putem pur și simplu să deconectăm și reconectăm firul de 5V sau firul de masă. De asemenea, este important să păstrăm Monitorul Serial Arduino deschis (Arduino IDE Menu > Tools > Serial Monitor) deoarece ne va oferi informații importante și ne va ajuta să depanezi schița ta.

USB to TTL Programmer



USB to TTL programmer connected to an Esp32-Cam (Source: Luca Dinale via All3DP)

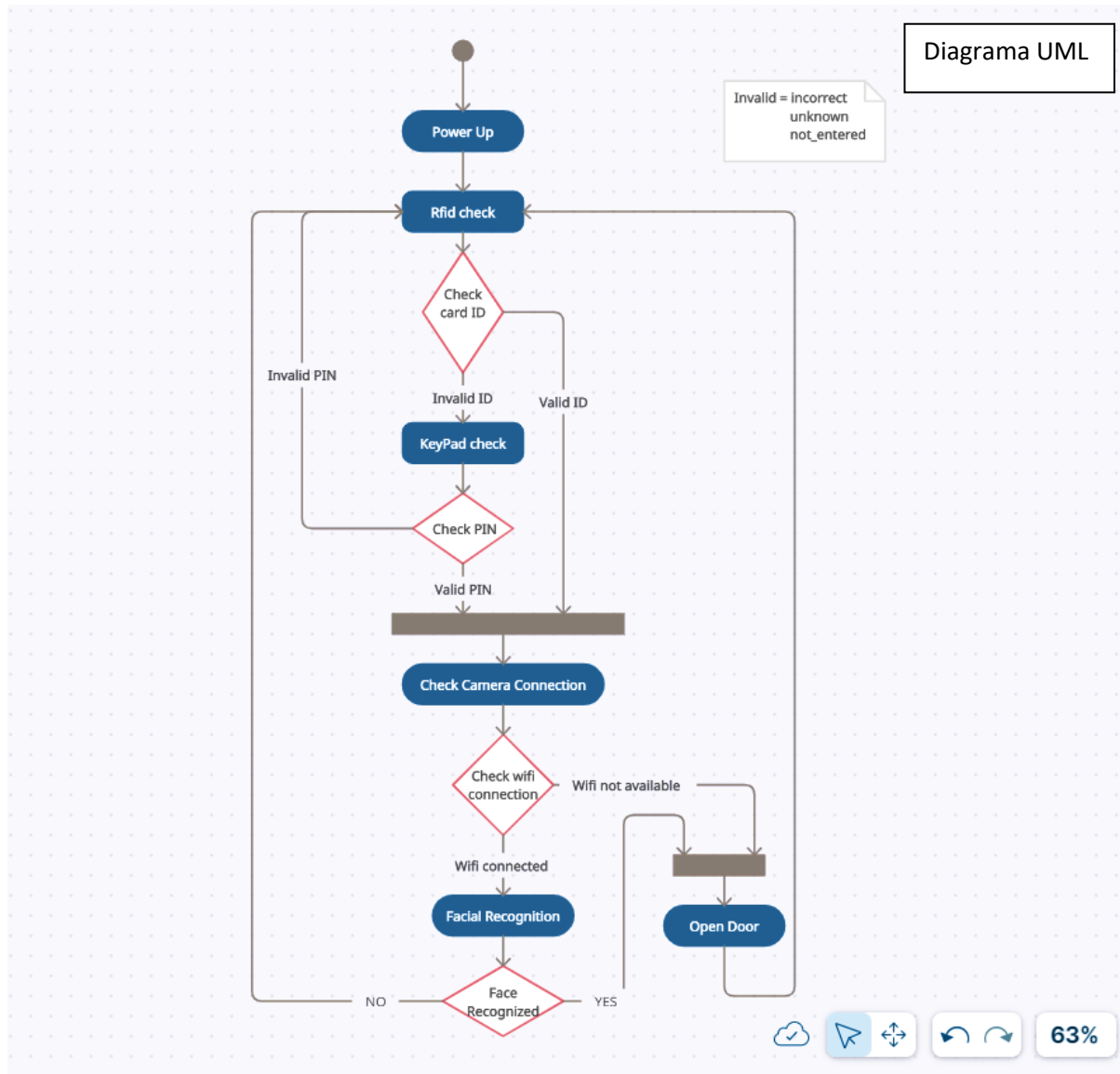
USB to TTL programmer are exact aceeași funcționalitate ca și cablul - este de asemenea un convertor de la USB la Serial, permițându-ți să-l conectezi pur și simplu la un port USB de pe computerul tău. Diferența dintre cablu și programator este dispunerea diferită a pinilor conexiunii seriale:

Conectează RX pe programatorul USB to TTL la UOT pe placă.

Conectează TX pe programatorul USB to TTL la UOR pe placă.

De obicei, USB to TTL programmers au un jumper pentru a selecta 3.3V sau 5V. Trebuie doar să asigurăm conectarea Esp32-Cam în mod corespunzător. Dacă setăm programmer-ul la 5V, ne conectăm la 5V pe placă; iar dacă setăm programmer-ul la 3.3V, ne conectăm la 3.3V pe placă.

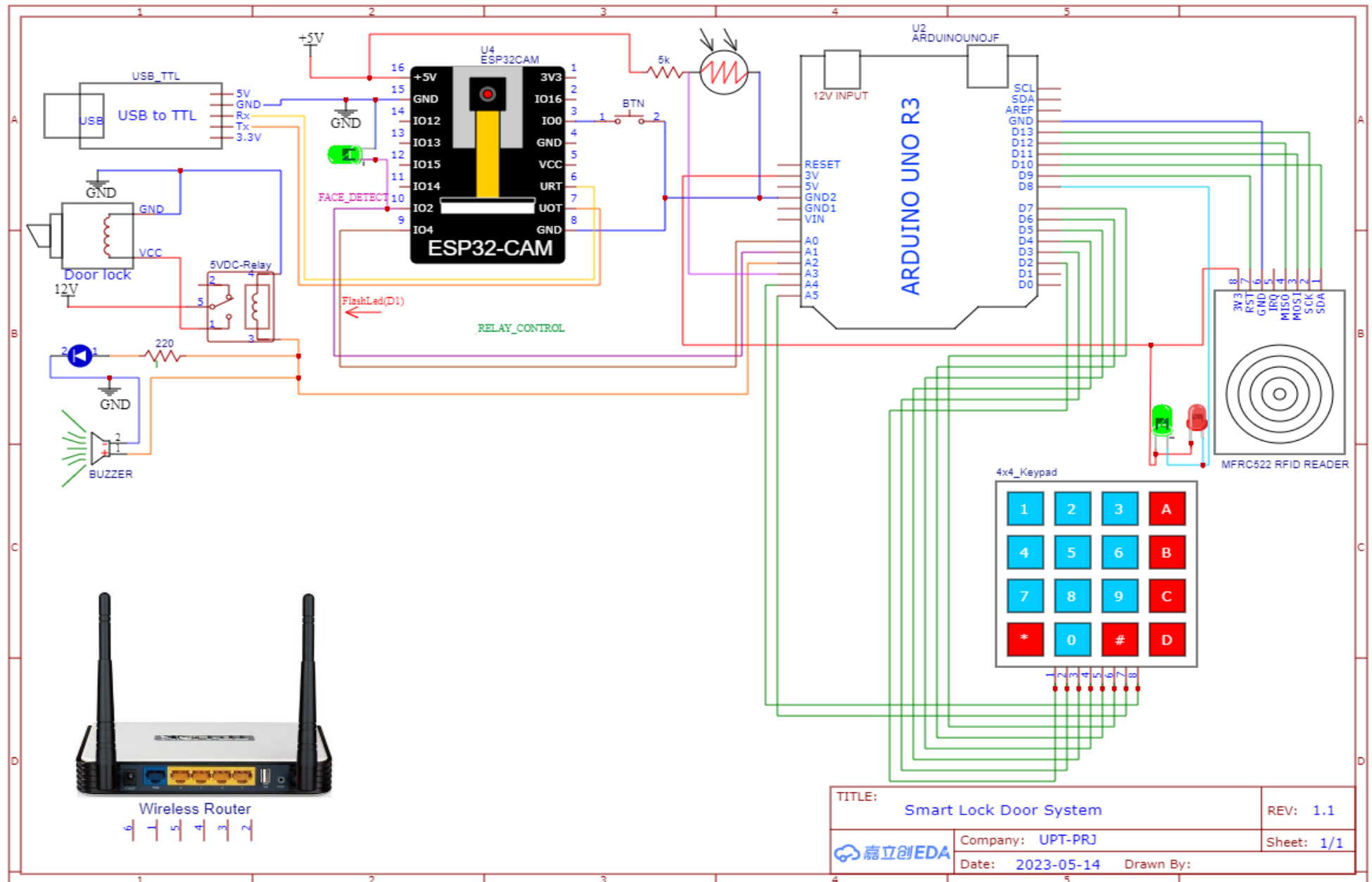
Funcționarea sistemului



Sistemul implementează o verificare a identității de tipul “*multiplepoint-authentication*”.

Odată alimentat sistemul trece în modul de verificare a tastaturii și a modulului rfid, dacă nici o operație nu detectează credențiale valide sistemul va parcurge iar ciclul de verificare. Dacă credențialele sunt valide (cardID, pin) sistemul va verifica starea modulului de cameră și wifi. În cazul în care nu se poate realiza conexiunea WLAN a modulului wifi sistemul va debloca ușa ca măsură preventivă în cazul în care există o problemă cu rețeaua, în caz contrar pentru un timp stabilit imaginile realizate de camera vor fi prelucrate folosind serverul web și în funcție de rezultatul comparației ușa se va deschide pentru un interval de timp sau se reia tot ciclul de funcționare.

Circuit design EasyEDA



Interconectarea componentelor

Modul Camera ESP32 -> Arduino

- Pinul IO2 al camerei transmite “1” logic (3,3v) când camera este în init mode sau când a realizat o recunoaștere facială, valoare citită în mod analog către Arduino.
- Pinul IO4 al camerei este legat la un pin digital de la Arduino și pornește LED-ul camerei când primește “1” logic, în condiții de luminiozitate scăzută.
- Pinul A3 al plăcii Arduino citește analog voltajul de la fotorezistorul de pe placa camerei, acesta fiind folosit ca senzor de lumină.

Tastatura -> Arduino

Pinii D7-D2 și A5-A4 sunt folosiți pentru scanarea tastelor apăsate și determinarea caracterelor introduse.

Modul rfid -> Arduino

Modulul comunică prin SPI cu placa Arduino, Arduino scanează continuu pentru a detecta prezența unui card și aprinderea ledului corespunzător validității acestuia (roșu-invalid, verde-valid).

Modulul de deblocare usa -> Arduino

Dacă s-au îndeplinit condițiile de deblocare Arduino trimite pe pinul A2 “1” logic acționând astfel releul 5VDC care funcționează ca un comutator și închide circuitul, alimentând actuatorul zăvorului ușii, rezultând în deblocarea acesteia, aprinderea unui LED și activarea unui buzzer ce emite un semnal acustic.

USB to TTL -> Camera ESP32

Folosind modulul USB to TTL încarcăm codul aferent camerei și server-ului web folosind o comunicare serială pe 2 canale RX și TX, folosind nivele logice compatibile (3,3V).

Cod sursă.Arduino

```

1 ///192.168.43.84///
2 #include <deprecated.h>
3 #include <MFRC522.h>
4 #include <MFRC522Extended.h>
5 #include <require_cppl.h>
6 #include <Keypad.h>
7 #include <SPI.h>
8 #include <MFRC522.h>
9
10 #define Password_Length 5
11 #define SS_PIN 10
12 #define RST_PIN 13
13 int RelayPin = A2;
14 int FlashPin = A0;
15 int FaceDetect = A1;
16 int RfidLeds = 8;
17 int parity = 0;
18
19 MFRC522 mfrc522(SS_PIN, RST_PIN);
20 char Data[Password_Length];
21 char Master[Password_Length] = "1111";
22 byte data_count = 0, master_count = 0;
23 bool Pass_is_good;
24 bool Rfid_is_good;
25 char customKey;
26 int i;
27 const byte ROWS = 4;
28 const byte COLS = 4;
29
30 char hexaKeys[ROWS][COLS] = {
31     {'1', '2', '3', 'A'},
32     {'4', '5', '6', 'B'},
33     {'7', '8', '9', 'C'},
34     {'*', '0', '#', 'D'}
35 };
36
37 byte rowPins[ROWS] = {7, 6, 5, 4}; //connect to t
38 byte colPins[COLS] = {3, 2, 18, 19}; //connect to
39
40 Keypad customKeypad = Keypad(makeKeymap(hexaKeys)
41

```

```

42 void setup() {
43     Serial.begin(9600);
44     SPI.begin();
45     mfrc522.PCD_Init();
46
47     pinMode(RelayPin, OUTPUT);
48     pinMode(RfidLeds, OUTPUT);
49     pinMode(FlashPin, OUTPUT);
50     Rfid_is_good = false;
51     Pass_is_good = false;
52
53     digitalWrite(RelayPin, LOW);
54     digitalWrite(RfidLeds, HIGH);
55     delay(500);
56     digitalWrite(RfidLeds, LOW);
57     delay(500);
58     digitalWrite(RfidLeds, HIGH);
59 }
60
61 void clearData() {
62     while (data_count != 0) {
63         Data[data_count--] = 0;
64     }
65     i=0;
66     return;
67 }
68
69 bool checkCard() {
70     if ( ! mfrc522.PICC_IsNewCardPresent()) {
71         return false;
72     }
73     if ( ! mfrc522.PICC_ReadCardSerial()) {
74         return false;
75     }
76     String content = "";
77     byte letter;
78     for (byte j = 0; j < mfrc522.uid.size; j++)
79     {
80         Serial.print(mfrc522.uid.uidByte[j] < 0x10 ? " 0" :
81         Serial.print(mfrc522.uid.uidByte[j], HEX);
82         content.concat(String(mfrc522.uid.uidByte[j] < 0x10
83         content.concat(String(mfrc522.uid.uidByte[j], HEX));

```

```

98 bool key() {
99     //if (Pass_is_good) return;
100     //if (checkCard() || Rfid_is_good) return
101     for(i=0; i<Password_Length; i++)
102     {
103         checkCard();
104         if(Rfid_is_good)
105         {
106             clearData();
107             return false;
108         }
109         customKey = customKeypad.getKey();
110         if (customKey)
111         {
112             Data[data_count] = customKey;
113             data_count++;
114         }else if (customKey == '#')
115         {
116             clearData();
117             return false;
118         }
119
120         if (data_count == Password_Length - 1)
121         {
122             if (!strcmp(Data, Master))
123             {
124                 Pass_is_good = true;
125                 Serial.print("Correct password ");
126                 clearData();
127                 return true;
128             }else
129             {
130                 Serial.print("Incorrect password ");
131                 Serial.println(Data);
132                 clearData();
133                 return false;
134             }
135         }
136     }
137     return false;
138 }

```

```

140 void Camera_Flash() {
141     int sensorValue = analogRead(A3);
142     float voltage = sensorValue * (5.0 / 1023.0);
143     if (voltage < 0.2) {
144         digitalWrite(FlashPin, HIGH);
145     }
146     else {
147         digitalWrite(FlashPin, LOW);
148     }
149     //Serial.println(voltage);
150 }
151
152 void loop()
153 {
154     int time = 5000;
155     float detect = analogRead(FaceDetect)*(5.0/1023.0);
156     key();
157     checkCard();
158     Camera_Flash();
159     if (Pass_is_good || Rfid_is_good)
160     {
161         while( (detect < 1.9) && (time > 0) )
162         {
163             time --;
164             delay(1);
165             Camera_Flash();
166             detect = analogRead(FaceDetect)*(5.0/1023.0);
167         }
168         if(detect>1.9)
169         {
170             digitalWrite(RelayPin, HIGH);
171             digitalWrite(RfidLeds, LOW);
172             Serial.println(" + face detected == unlock door");
173             delay(3000);
174             digitalWrite(RelayPin, LOW);
175             digitalWrite(RfidLeds, HIGH);
176         }
177         Pass_is_good = false;
178         Rfid_is_good = false;
179     }
180 }
181 }

```


Cod Sursă.CameraESP-32

[4]

```
void handle_message(WebsocketsClient &client, WebsocketsMessage msg)
{
    if (msg.data() == "stream") {
        g_state = START_STREAM;
        client.send("STREAMING");
    }
    if (msg.data() == "detect") {
        g_state = START_DETECT;
        client.send("DETECTING");
    }
    if (msg.data().substring(0, 8) == "capture:") {
        g_state = START_ENROLL;
        char person[FACE_ID_SAVE_NUMBER * ENROLL_NAME_LEN] = {0,};
        msg.data().substring(8).toCharArray(person, sizeof(person));
        memcpy(st_name.enroll_name, person, strlen(person) + 1);
        client.send("CAPTURING");
    }
    if (msg.data() == "recognise") {
        g_state = START_RECOGNITION;
        client.send("RECOGNISING");
    }
    if (msg.data().substring(0, 7) == "remove:") {
        char person[ENROLL_NAME_LEN * FACE_ID_SAVE_NUMBER];
        msg.data().substring(7).toCharArray(person, sizeof(person));
        delete_face_id_in_flash_with_name(&st_face_list, person);
        send_face_list(client); // reset faces in the browser
    }
    if (msg.data() == "delete_all") {
        delete_all_faces(client);
    }
}

void open_door(WebsocketsClient &client) {
    if (digitalRead(relay_pin) == LOW) {
        digitalWrite(relay_pin, HIGH); //close (energise) relay so door un
        Serial.println("Door Unlocked");
        client.send("door_open");
        door_opened_millis = millis(); // time relay closed and door opens
    }
}
```

Referinte:

- [1] <https://docs.arduino.cc/hardware/uno-rev3/>*
- [2] <https://www.circuitbasics.com/how-to-set-up-a-keypad-on-an-arduino/>*
- [3] <https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/>*
- [4] <https://all3dp.com/2/esp32-cam-arduino-tutorial/>*
- [5] <https://www.circuito.io/blog/arduino-uno-pinout/>*
- [6] <https://learn.sparkfun.com/tutorials/serial-peripheral-interface-spi/all/>*
- [7] <https://ocw.cs.pub.ro/courses/pm/lab/lab3/>*
- [8] <https://github.com/robotzero1/esp32cam-access-control>*