Міністерство освіти і науки України
Харківський національний університет радіоелектроніки
Факультет Комп'ютерної інженерії та управління
(повна назва)
Кафедра Автоматизації проєктування обчислювальної техніки
(повна назва)

КВАЛІФІКАЦІЙНА РОБОТА
Пояснювальна записка

рівень вищої освіти перший (бакалаврський)

Розумний електронний замок із захистом на основі Arduino
(тема)

Виконав: здобувач 4 курсу,
групи КІУКІ-21-7
Донець В.В
(прізвище, ініціали)

спеціальності 123 – Комп'ютерна інженерія
(шифр і назва спеціальності)

Тип програми освітньо-професійна
(освітньо-професійна або освітньо-наукова)

Освітня програма Комп'ютерна інженерія
(повна назва освітньої програми)

Керівник ас. Хаханов І.В.
(посада, прізвище, ініціали)

Допускається до захисту

Зав. кафедри _____ Чумаченко С.В.
(підпис) (прізвище, ініціали)

2025 р.

Харківський національний університет радіоелектроніки

Факультет Комп'ютерної інженерії та управління

Кафедра Автоматизації проєктування обчислювальної техніки

Рівень вищої освіти перший (бакалаврський)

Спеціальність 123 Комп'ютерна інженерія

Тип програми Освітньо-професійна

Освітня програма Комп'ютерна інженерія

ЗАТВЕРДЖУЮ:
Зав. кафедри
_____
(підпис)
«____»_____ 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ

Здобувачеві Донцю Віталію Вікторовичу
(прізвище, ім'я, по батькові)

1. Тема роботи Розумний електронний замок із захистом на основі Arduino

затверджена наказом університету від 21.05.2025 р. № 403Ст

2. Термін подання студентом роботи до екзаменаційної комісії _____2025 р.

3. Вихідні дані до роботи
Мікроконтролер Arduino Nano.
Електромеханічний соленоїдний замок.

4. Перелік питань, що потрібно опрацювати в роботі
Аналіз існуючих систем контролю доступу та їх класифікація.
Вибір апаратної платформи та компонентів для реалізації системи.
Розробка структурної та функціональної схеми пристрою.
Програмна реалізація алгоритмів автентифікації (PIN-код, RFID).
Інтеграція та налаштування компонентів системи.
Тестування роботи пристрою та аналіз результатів.

5. Перелік графічного матеріалу із зазначенням креслеників, схем, плакатів, комп'ютерних ілюстрацій (п.5 включається до завдання за рішенням випускової кафедри) _____
13 слайдів

КАЛЕНДАРНИЙ ПЛАН

№	Назва етапів роботи	Терміни виконання етапів роботи	Примітка
1	Видача теми проєкту, узгодження і затвердження теми	06.05.2025 – 08.05.2025	
2	Аналіз предметної області, вибір компонентів системи	08.05.2025 – 13.05.2025	
3	Розробка структурної схеми пристрою, вибір апаратної платформи	13.05.2025 – 18.05.2025	
4	Розробка функціональної схеми програми	18.05.2025 – 20.05.2025	
5	Розробка програмних модулів. Проведення тестування	20.05.2025 – 28.05.2025	
6	Оформлення пояснювальної записки	28.05.2025 – 06.06.2025	
7	Підготовка ілюстративних матеріалів.	06.06.2025 – 12.06.2025	

Дата видачі завдання 06.05.2025 р.

Здобувач _____
(підпис)

Керівник роботи _____ ас. Хаханов І.В.
(підпис) (посада, прізвище, ініціали)

РЕФЕРАТ

Пояснювальна записка кваліфікаційної роботи містить 48 сторінок, 12 рисунків, 10 джерел посилання.

ARDUINO NANO, C++, RFID, ARDUINO IDE, MFRC522, SPI, РЕЛЕ, OLED, ЗАМОК, КЛАВІАТУРА.

Метою кваліфікаційної роботи є розробка розумного електронного замка з надійною системою захисту від несанкціонованого доступу. У роботі проведено аналіз існуючих методів забезпечення безпеки електронних замків, визначено їх переваги та недоліки, а також виявлено потенційні проблеми, що можуть виникнути під час впровадження таких систем. Програмна частина реалізована на мові C++ з використанням апаратної платформи Arduino. Для створення пристрою було використано такі компоненти, як Arduino Nano, електромеханічний соленоїдний замок, матрична клавіатура та RFID-модуль RC522. Розроблена система підтримує автентифікацію користувача як через введення PIN-коду, так і за допомогою RFID-картки чи мітки. Проведено експериментальне тестування пристрою, яке підтвердило його ефективність у забезпеченні безпеки та здатність запобігати несанкціонованому доступу.

ABSTRACT

The explanatory note of the qualification work contains 48 pages, 12 figures, 10 references.

ARDUINO NANO, C++, RFID, ARDUINO IDE, MFRC522, SPI, RELAY, OLED, LOCK , KEYBOARD.

The purpose of the qualification work is to develop a smart electronic lock with a reliable system of protection against unauthorized access. The work analyzes existing methods of ensuring the security of electronic locks, identifies their advantages and disadvantages, and identifies potential problems that may arise during the implementation of such systems. The software part is implemented in C++ using the Arduino hardware platform. To create the device, we used components such as the Arduino Nano, an electromechanical solenoid lock, a matrix keyboard, and the RC522 RFID module. The developed system supports user authentication both by entering a PIN code and by using an RFID card or tag. Experimental testing of the device was conducted, which confirmed its effectiveness in ensuring security and the ability to prevent unauthorized access.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ, СИМВОЛІВ, ОДИНИЦЬ, СКОРОЧЕНЬ І ТЕРМІНІВ

RFID (Radio Frequency Identification) — технологія безконтактної ідентифікації, що використовує радіочастотні сигнали для передачі даних.

MFRC522 — RFID-зчитувач, який працює на частоті 13.56 МГц і підтримує стандарт MIFARE.

PIN-код (Personal Identification Number) — персональний ідентифікаційний номер, який використовується для автентифікації користувача.

SPI (Serial Peripheral Interface) — інтерфейс для підключення периферійних пристроїв до мікроконтролера.

I2C (Inter-Integrated Circuit) — двовивідний інтерфейс для з'єднання мікросхем.

OLED (Organic Light Emitting Diode) — технологія дисплеїв, що використовує органічні світловипромінювальні діоди для створення зображень.

Relay Module — релейний модуль, що дозволяє використовувати електричні сигнали для управління високовольтними пристроями.

Fail-secure — режим, при якому система залишається заблокованою при відсутності живлення.

Fail-safe — режим, при якому система відкривається або залишається в безпечному стані при відключенні живлення.

UART (Universal Asynchronous Receiver-Transmitter) — стандартний інтерфейс для асинхронного обміну даними.

UID (Unique Identifier) — унікальний ідентифікатор, що присвоюється RFID-карткам для ідентифікації користувачів.

DAC (Discretionary Access Control) – модель контролю доступу, де власник ресурсу визначає, хто має до нього доступ, надаючи дозволи на

індивідуальній основі.

RBAC (Role-Based Access Control) – модель контролю доступу, де дозволи призначаються ролям, а користувачам – ролі, відповідно до їхніх функцій в організації.

ABAC (Attribute-Based Access Control) – модель контролю доступу, що приймає рішення про доступ на основі атрибутів суб'єкта (користувача), об'єкта (ресурсу) та середовища.

ВСТУП

У сучасному світі забезпечення безпеки житла та комерційних приміщень стає дедалі важливішим завданням. Одним із основних елементів безпеки є ефективна система контролю доступу, яка дозволяє забезпечити зручність, швидкість і надійність відкриття дверей, знижуючи ризик несанкціонованого проникнення. Традиційні механічні замки вже не відповідають вимогам сучасності, тому необхідність у розробці розумних електронних замків стає очевидною.

Цей проект присвячений розробці розумного електронного замка, що працює на основі мікроконтролера Arduino. Система передбачає два основні способи підтвердження доступу: зчитування RFID-картки та введення PIN-коду через матричну клавіатуру. У разі кількох невдалих спроб введення неправильних даних, система автоматично блокує доступ на певний час, забезпечуючи додатковий рівень захисту. Прилад оснащений OLED-дисплеєм для відображення статусу системи та індикації введення PIN-коду або сканування RFID-карти.

Основою роботи замка є електромеханічний соленоїдний замок, який забезпечує фізичне відкриття дверей після підтвердження доступу. Проект реалізовано з урахуванням потреби в надійності, простоті користування та швидкому доступі, без необхідності подальших удосконалень.

Метою цього проекту є створення доступного та ефективного рішення для контролю доступу, яке поєднує простоту в користуванні, високу надійність та гнучкість у налаштуванні під потреби кінцевого користувача.

1АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Системи контролю доступу

Системи контролю доступу (СКД) є основною складовою безпеки, що застосовується для регулювання прав доступу до ресурсів як у фізичному, так і в обчислювальному середовищі[1]. Головною метою СКД є захист важливих даних та ресурсів від несанкціонованого доступу, гарантуючи, що лише авторизовані особи чи системи можуть отримувати доступ до конкретних ресурсів у встановлених умовах. Завдяки цьому СКД забезпечують конфіденційність, цілісність і доступність критичних ресурсів.

Контроль доступу базується на трьох основних процесах: ідентифікації, автентифікації та авторизації (рисунок 2.1).

Рисунок 1.1 – Основні процеси контролю доступу

На базовому рівні контроль доступу може реалізовуватись через паролі та права користувачів. Проте сучасні системи часто використовують більш складні технології, такі як біометрична ідентифікація, багатофакторна автентифікація та динамічні політики доступу на основі атрибутів або правил[1]. СКД виконують роль охоронця, перевіряючи право доступу, а

також постійно моніторять діяльність, забезпечуючи дотримання політик доступу та реєструючи всі події для подальшого аналізу і виявлення потенційних загроз безпеці[2].

1.1.1 Класифікація СКД

Системи контролю доступу можна класифікувати за кількома ознаками, серед яких найбільш важливими є спосіб управління пристроями та масштаб системи. За способом управління пристроями СКД поділяються на автономні (локальні) системи, мережеві (централізовані) системи та універсальні системи. Автономні системи призначені для управління одним або кількома пристроями, наприклад, дверима, без необхідності передавати інформацію на центральний пульт або отримувати контроль від оператора. Всі дані про користувачів та логіка прийняття рішень зберігаються безпосередньо в контролері пристрою. Такі системи можуть також збирати дані для подальшого аналізу. У той час мережеві системи призначені для управління великою кількістю точок доступу з обміном інформацією з центральним сервером або пультом, що дозволяє оператору контролювати і керувати системою в режимі реального часу. Ці системи часто інтегруються з іншими засобами безпеки, такими як відеоспостереження або охоронно-пожежна сигналізація. Універсальні системи поєднують функції автономних та мережевих систем, дозволяючи працювати в мережевому режимі під управлінням центрального пристрою, а в разі збоїв мережевого обладнання переходити в автономний режим[3].

За масштабом, тобто за кількістю точок доступу та користувачів, СКД поділяються на малі, середні та великі системи. Малі системи використовуються для управління одиничними точками доступу, наприклад, у невеликих офісах. Середні системи контролюють десятки точок доступу та тисячі користувачів, наприклад, у банках, підприємствах або готелях. Великі системи обслуговують сотні точок доступу та десятки тисяч користувачів, їх

використовують на великих промислових підприємствах, аеропортах та інших масштабних об'єктах. Крім того, існують класифікації за типом ідентифікації, наприклад, біометричні СКД або системи, що використовують картки для доступу[3].

1.1.2 Основні компоненти СКД

Будь-яка система контролю доступу складається з кількох основних компонентів, які взаємодіють для забезпечення її ефективної роботи. Основними компонентами СКД є ідентифікатори, зчитувачі, контролери та виконавчі пристрої[3].

Ідентифікатори – це засоби, за допомогою яких користувач підтверджує свою особу. Вони можуть бути такими, що користувач знає (наприклад, пароль або PIN-код), має (наприклад, карту доступу, брелок чи смартфон) або тим, чим він є (наприклад, біометричні дані, такі як відбиток пальця чи райдужка ока)[3]. Поширеними фізичними ідентифікаторами є RFID-карти стандартів Em-Marine та MIFARE, а також брелоки.

Зчитувачі – це пристрої, які зчитують інформацію з ідентифікаторів і передають її на контролер[3]. Тип зчитувача залежить від виду ідентифікатора: RFID-зчитувачі для карт і брелоків, клавіатури для введення PIN-кодів, біометричні сканери для зчитування відбитків пальців чи інших біометричних даних.

Контролери – це мозок СКД. Контролер отримує дані від зчитувача, аналізує їх на основі закладеної логіки та бази даних авторизованих користувачів і приймає рішення про надання або відмову в доступі. Контролери зберігають конфігурацію системи, режими роботи та список осіб з правами доступу. Вони можуть бути автономними, обслуговуючи одну точку доступу, або мережевими, працюючи під управлінням комп'ютера в більших системах[4]. У розробленому проекті роль контролера виконує мікроконтролер Arduino.

Виконавчі пристрої – це фізичні бар'єри, які безпосередньо обмежують або дозволяють доступ. До них відносяться електронні замки (електромеханічні, електромагнітні), турнікети, шлагбауми, автоматичні ворота тощо[3]. У цьому проекті виконавчим пристроєм є електронний замок, керований через реле.

1.1.3 Моделі контролю доступу

Існує кілька моделей контролю доступу, які визначають, як приймаються рішення щодо надання дозволу на доступ, одним з таких є обов'язковий контроль доступу або MAC, він регулюється центральним органом на основі міток безпеки, які призначають допуски для суб'єктів та класифікації для об'єктів. Користувачі не мають можливості змінювати дозволи. Ця модель є однією з найбільш обмежувальних і використовується в середовищах з високими вимогами до безпеки, таких як урядові та військові установи[1].

Дискреційний контроль доступу (DAC) дає можливість власнику ресурсу визначати, хто має до нього доступ. Це більш гнучка модель, яка дозволяє користувачам надавати доступ до ресурсів на індивідуальній основі[1].

Контроль доступу на основі ролей (RBAC) передбачає, що дозволи на доступ призначаються ролям, а користувачам — ролі, залежно від їхніх обов'язків або функцій в організації. Ця модель є однією з найпоширеніших, оскільки вона чітко відображає бізнес-операції і є гнучкою[1].

Контроль доступу на основі правил забезпечує доступ або його заборону на основі набору попередньо визначених правил та політик, таких як час доби або місцезнаходження[1].

Контроль доступу на основі атрибутів (ABAC) приймає рішення про доступ на основі атрибутів суб'єкта (користувача), об'єкта (ресурсу) та середовища. Це забезпечує детальніший та контекстно-залежний контроль[1].

Розроблюваний в рамках цієї роботи електронний замок реалізує просту форму контролю доступу, що ґрунтується на ідентифікації через конкретний UID RFID-карти або введення PIN-коду. Це можна вважати елементарною реалізацією контролю доступу за ідентичністю, що має риси дискреційної моделі, де адміністратор системи визначає дозволений UID та PIN, та моделі на основі правил (доступ надається, якщо UID/PIN збігаються з збереженими).

1.1.4 Сфери застосування СКД

Системи контролю доступу мають широке застосування в різних сферах для забезпечення захисту фізичних та інформаційних активів. У комерційних та державних установах, таких як офісні будівлі, банки та урядові органи, СКД використовуються для обмеження доступу до приміщень, кабінетів з підвищеним рівнем безпеки та сховищ даних. На промислових об'єктах, таких як заводи та підприємства, вони застосовуються для контролю доступу на територію, до виробничих цехів, складів, а також для обліку робочого часу. Транспортні вузли, включаючи аеропорти та вокзали, використовують СКД для розмежування зон доступу для пасажирів і персоналу.

У закладах освіти та охорони здоров'я, таких як школи, університети та лікарні, системи контролю доступу застосовуються для контролю доступу до навчальних аудиторій, лабораторій, палат пацієнтів та архівів медичних записів. У житловому секторі багатоквартирні будинки, приватні будинки та котеджні містечка оснащуються електронними замками та СКД для підвищення безпеки мешканців. У сфері інформаційних технологій СКД є важливим елементом захисту даних, хмарних середовищ, програмних додатків та комп'ютерних систем від несанкціонованого доступу і кібератак. Системи можуть бути як простими автономними замками на окремі двері, так і складними мережевими системами, що охоплюють цілий комплекс будівель та інтегруються з іншими системами безпеки.

1.2 Принципи роботи електронних замків

Основним принципом роботи електронного замка є керування доступом за допомогою електричного сигналу, який активує або деактивує замикаючий механізм. На відміну від механічних замків, що потребують фізичного ключа для взаємодії з штифтами або сувальдами, електронні замки використовують різноманітні методи ідентифікації для надання доступу. Це може бути введення PIN-коду, прикладання RFID-карти чи брелка, сканування біометричних даних (наприклад, відбитка пальця) або команда зі смартфона[5].

Ключові відмінності електронних замків від механічних полягають у наступному:

– спосіб керування: механічні замки керуються фізичним ключем, тоді як електронні – електричними сигналами, що генеруються на основі різних методів ідентифікації;

– функціональність: електронні замки часто пропонують додаткові можливості, такі як дистанційне керування, програмування тимчасового доступу, ведення журналу подій (хто і коли відкривав замок), а також інтеграцію з системами "розумного будинку" чи охоронними системами;

– безпека: електронні замки можуть забезпечувати вищий рівень безпеки, особливо моделі без традиційної замкової щілини (замки-невидимки) або ті, що використовують багатофакторну автентифікацію. Проте їх безпека залежить також від стійкості до електронних методів злому[6];

– залежність від живлення: електронні замки потребують джерела живлення, на відміну від механічних, які не залежать від енергії. У разі відключення живлення поведінка електронного замка може бути різною в залежності від конструкції: у моделі fail-safe замок відкривається, а у fail-secure залишається закритим.

1.2.1 Типи замикаючих механізмів

Електронні замки можуть використовувати різні типи замикаючих механізмів, які приводяться в дію електричним сигналом:

Електромеханічні замки поєднують традиційний механічний ригельний механізм з електричним приводом, таким як соленоїд або невеликий мотор, що втягує або висуває ригель. Ці замки часто мають можливість резервного відкриття за допомогою механічного ключа. При зачиненні дверей ригель може стиснути пружину, а електричний сигнал на соленоїд звільняє фіксатор пружини, що дозволяє втягути ригель[5].

Електромагнітні замки складаються з потужного електромагніту, встановленого на дверній рамі, та металевої пластини (якоря) на дверному полотні. Коли подається живлення, електромагніт притягує пластину, утримуючи двері закритими. Відсутність рухомих механічних частин забезпечує високу зносостійкість. Такі замки зазвичай є fail-safe, тобто вони відкриваються при відключенні живлення.

Електроригельні замки використовують висувний металевий ригель, який рухається за допомогою електроприводу і входить у відповідну частину на дверній коробці або полотні, забезпечуючи надійне замикання[5].

Електрозащіпки встановлюються на дверну раму замість стандартної відповідної планки. При подачі сигналу язичок защіпки звільняється, дозволяючи відкривати двері, які зазвичай фіксуються механічним замком, наприклад, фалевою клямкою.

Електромоторні замки використовують електродвигун для переміщення ригелів, що забезпечує плавне і потужне замикання та відмикання дверей.

Вибір типу замикаючого механізму залежить від типу дверей, вимог до безпеки та умов експлуатації.

2 ВИБІР КОМПОНЕНТІВ СИСТЕМИ

2.1 Вибір мікроконтролера та основних компонентів

Розробка розумного електронного замка вимагає ретельного вибору компонентів, що гарантуватимуть надійну та ефективну роботу системи. Основними критеріями для вибору стали доступність, ціна, легкість інтеграції та наявність програмної підтримки для кожного з компонентів. У цьому проекті центральним елементом є мікроконтролер Arduino, до якого підключені модулі для ідентифікації користувача, відображення даних та управління виконавчим механізмом замка.

2.1.1 Мікроконтролер Arduino

Для реалізації розумного електронного замка як основний керуючий пристрій була обрана платформа Arduino (рисунок 2.1).

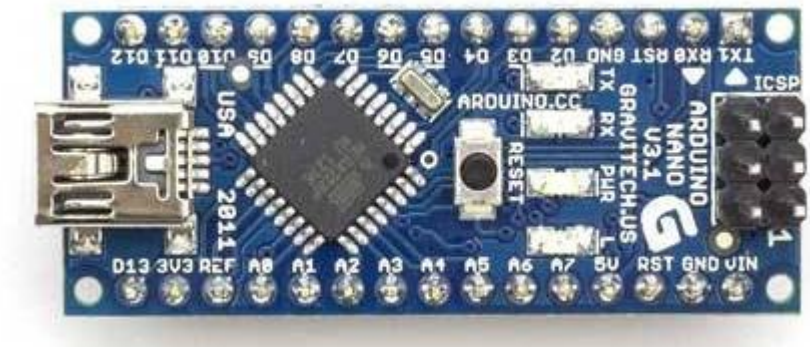

Рисунок 2.1 – плата Arduino Nano

Цей вибір обумовлений кількома важливими факторами. По-перше, Arduino є відкритою апаратною платформою, що надає велику кількість доступних плат та модулів розширення, що робить її економічно вигідним

рішенням для прототипування та розробки. По-друге, простота програмування за допомогою середовища Arduino IDE та мови C/C++, заснованої на фреймворку Wiring, дозволяє швидко реалізовувати необхідну логіку навіть розробникам з базовим рівнем знань. По-третє, велика і активна спільнота розробників забезпечує наявність численних бібліотек для роботи з різноманітними сенсорами та модулями, що значно прискорює процес розробки.

У рамках цього проекту плата Arduino Nano, побудована на мікроконтролері ATmega328P, виконує всі основні функції керування замком. Вона відповідає за ініціалізацію та опитування підключених периферійних пристроїв, таких як RFID-зчитувач, матрична клавіатура, OLED-дисплей, зумер та релейний модуль. Arduino обробляє дані, отримані від RFID-зчитувача (UID картки) та клавіатури (PIN-код), порівнює їх з попередньо збереженими авторизованими значеннями і приймає рішення щодо надання або відмови в доступі на основі перевірки.

Крім того, мікроконтролер керує виконавчим механізмом замка через релейний модуль, відображає актуальну інформацію на OLED-дисплеї та генерує звукові сигнали для індикації різних подій. Він також реалізує логіку безпеки, включаючи блокування системи після кількох невдалих спроб доступу та встановлення таймауту неактивності.

Arduino має достатню кількість цифрових і аналогових пінів вводу/виводу та підтримує необхідні інтерфейси зв'язку, такі як SPI для RFID-модуля та I2C для OLED-дисплея, що робить її оптимальним вибором для цього проекту з урахуванням функціональних вимог та складності реалізації.

2.1.2 Модуль RFID-зчитувача MFRC522

Для реалізації безконтактної ідентифікації за допомогою RFID-карт був обраний модуль MFRC522 (рисунок 2.2), який є популярним та доступним рішенням для проектів на Arduino. Цей модуль працює на частоті 13.56 МГц

та підтримує роботу з різними типами карт сімейства MIFARE, такими як MIFARE Classic, MIFARE Ultralight, MIFARE Pro та MIFARE DESFire[7].

Рисунок 2.2 – Модуль RFID RC522 з ключ картою та брелоком MIFARE

У проекті модуль MFRC522 виконує основні функції зчитування унікального ідентифікатора (UID) RFID-карти. Коли карта підноситься до антени зчитувача, модуль отримує її UID, який потім передається на мікроконтролер Arduino для подальшої обробки. Для реалізації цього процесу використовується бібліотека MFRC522.h, що надає всі необхідні функції для ініціалізації зчитувача, перевірки наявності картки в полі дії та зчитування її серійного номера.

Для забезпечення безпеки у проекті зберігається один дозволений UID карти, і при зчитуванні UID з картки, він порівнюється з попередньо збереженим значенням для прийняття рішення про надання доступу. Підключення модуля до Arduino здійснюється через інтерфейс SPI, що дозволяє швидко та ефективно обмінюватися даними.

Для кращого розуміння внутрішньої структури модуля MFRC522, на рисунку 2.3 представлена його спрощена блок-схема, яка демонструє основні

компоненти та їх взаємодію для здійснення безконтактної ідентифікації.

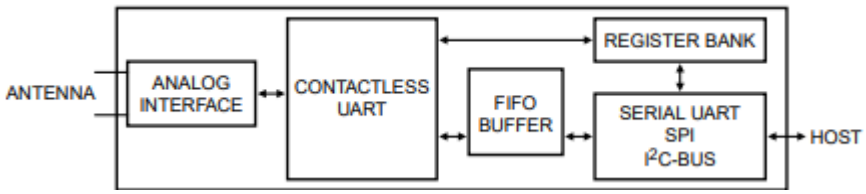

Рисунок 2.3 – Спрощена блок-схема внутрішньої структури MFRC522

Модуль MFRC522 використовує антену для безконтактного зчитування RFID карток. Коли картка наближається до антени, вона генерує електричний сигнал, що містить унікальний ідентифікатор (UID). Чип MFRC522 приймає цей сигнал, обробляє його і передає зчитану інформацію в мікроконтролер через інтерфейс SPI. Дані передаються з допомогою FIFO буфера, що тимчасово зберігає інформацію перед її подальшою обробкою. Мікроконтролер приймає отриманий UID, порівнює його з попередньо збереженим значенням та приймає рішення про надання або відмову в доступі. Цей процес є основою роботи модуля MFRC522 в системах безконтактної ідентифікації[7].

2.1.3 Матрична клавіатура

Для введення PIN-коду як додаткового методу автентифікації використовується матрична клавіатура 4x4, яка складається з 16 кнопок (рисунок 2.4). Це популярний компонент для введення даних у вбудованих системах, який дозволяє користувачеві вводити числові комбінації або інші символи[8].

Рисунок 2.4 – Матрична клавіатура

У проекті клавіатура використовується для введення 4-значного PIN-коду, що є необхідним для доступу до системи. Після натискання кнопок на клавіатурі інформація передається на мікроконтролер Arduino для подальшої обробки. У програмному коді для зчитування натиснутих клавіш застосовується бібліотека Keypad.h, яка спрощує процес опитування клавіатури.

Клавіатура має розкладку з чотирьох рядків та чотирьох стовпців, і кожна клавіша асоціюється з певним символом або цифрою. Після введення PIN-коду, програма порівнює його з еталонним значенням, що збережене в системі. Для підвищення конфіденційності під час введення код відображаються символи-заповнювачі (зірочки *).

2.1.4 OLED-дисплей SSD1306

Для візуального відображення інформації та взаємодії з користувачем в проекті використовується монохромний OLED-дисплей на базі контролера

SSD1306 (рисунок 2.5). Дисплей, має діагональ 0.96 дюйма та роздільну здатність 128x64 пікселів, він є популярним рішенням для проектів на Arduino завдяки високій контрастності, низькому енергоспоживанню та простоті підключення через інтерфейс I2C[9].

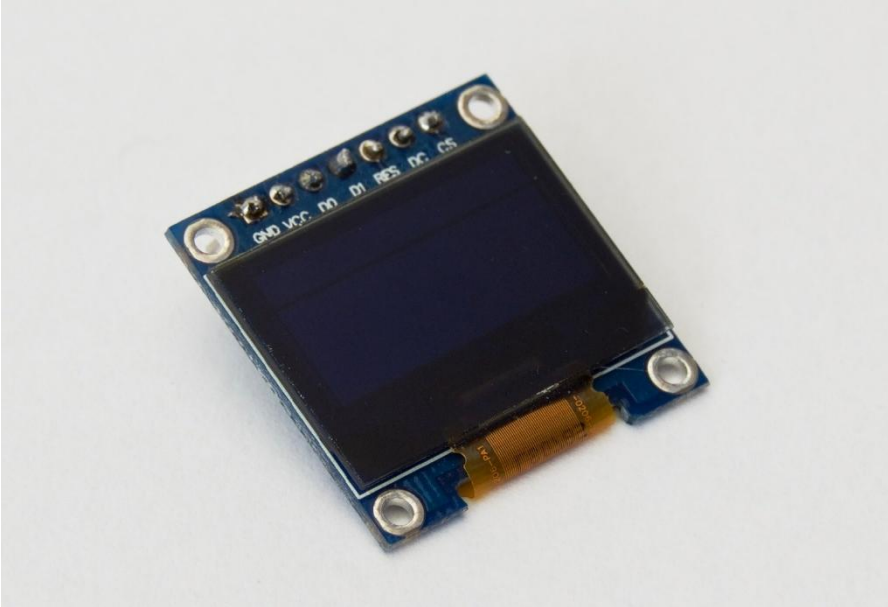

Рисунок 2.5 – SSD1306 OLED дисплей

У даному проекті OLED-дисплей виконує кілька важливих функцій. Він відображає стан системи, включаючи привітальний екран, запити на введення даних, повідомлення про успішний або невдалий доступ, а також індикацію стану блокування системи. Дисплей також забезпечує взаємодію з користувачем, надаючи візуальні підказки та зворотний зв'язок під час введення PIN-коду або перевірки RFID картки.

Для роботи з дисплеєм у програмному коді використовуються бібліотеки Adafruit_SSD1306.h та Adafruit_GFX.h, що значно полегшують взаємодію з цим компонентом. Дисплей підключається до мікроконтролера через інтерфейс I2C з адресою 0x3C, вказаною при ініціалізації за допомогою display.begin(SSD1306_SWITCHCAPVCC, 0x3C). Для апаратного скидання дисплея використовується пін OLED_RESET (пін 4).

Дисплей виводить різноманітну текстову інформацію для зручності використання, а також показує таймер зворотного відліку під час блокування системи. Це дозволяє користувачеві отримувати чітку і зрозумілу інформацію про поточний стан замка та процес автентифікації.

2.1.5 Релейний модуль

Для керування виконавчим механізмом електронного замка, таким як соленоїдний або електромеханічний замок, який потребує більшої напруги та струму, ніж може забезпечити пін мікроконтролера, використовуються релейні модулі (рисунок 2.6). Реле функціонує як електрично керований перемикач, що дозволяє низьковольтному сигналу від Arduino комутувати силове коло живлення замка.

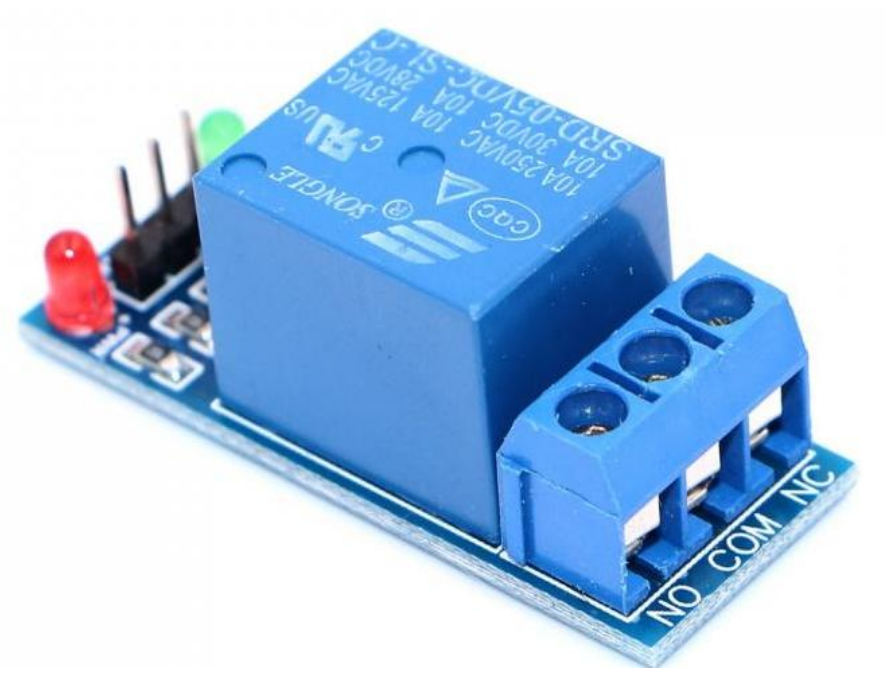

Рисунок 2.6 - Модуль реле 5В 10А низького рівня

Цей релейний модуль активується низьким рівнем сигналу, що є типовим для реле з низьким рівнем активації. Коли пін мікроконтролера подає

сигнал низького рівня на реле, воно замикає електричне коло, подаючи живлення на виконавчий механізм замка і забезпечуючи його відкриття[10]. Коли сигнал високого рівня припиняється, реле деактивується, розмикаючи коло і закриваючи замок. Зазначений тип реле з нормально розімкненим контактом (NO) означає, що замок залишається закритим, якщо реле не отримує сигналу або знаходиться в стані відкриття.

Модуль реле 5В 10А низького рівня є оптимальним вибором для проектів на базі Arduino завдяки простоті підключення та здатності керувати великими навантаженнями при низькому споживанні енергії.

2.2 Опис електронної схеми

На рисунку 2.7 подано електричну схему підключення всіх компонентів розумного електронного замка. Центральним елементом є плата Arduino Nano. До її цифрових пінів 2, 3, 4, 5 підключені виводи рядків матричної клавіатури, а до пінів 6, 7, 8, A0 – виводи стовпців. Модуль RFID MFRC522 підключений до апаратного SPI-інтерфейсу Arduino (піни 10, 11, 12, 13) та до піна 9 для скидання. OLED-дисплей SSD1306 підключений до I2C-інтерфейсу Arduino (піни A4, A5) та до піна 4 для скидання. Пасивний зумер підключений до цифрового піна 5 та GND. Релейний модуль керується сигналом з піна A1 Arduino; його силова частина комутує окреме коло живлення для електрозамка. Всі компоненти, що живляться від Arduino, підключені до відповідних виводів 5V/3.3V та GND.

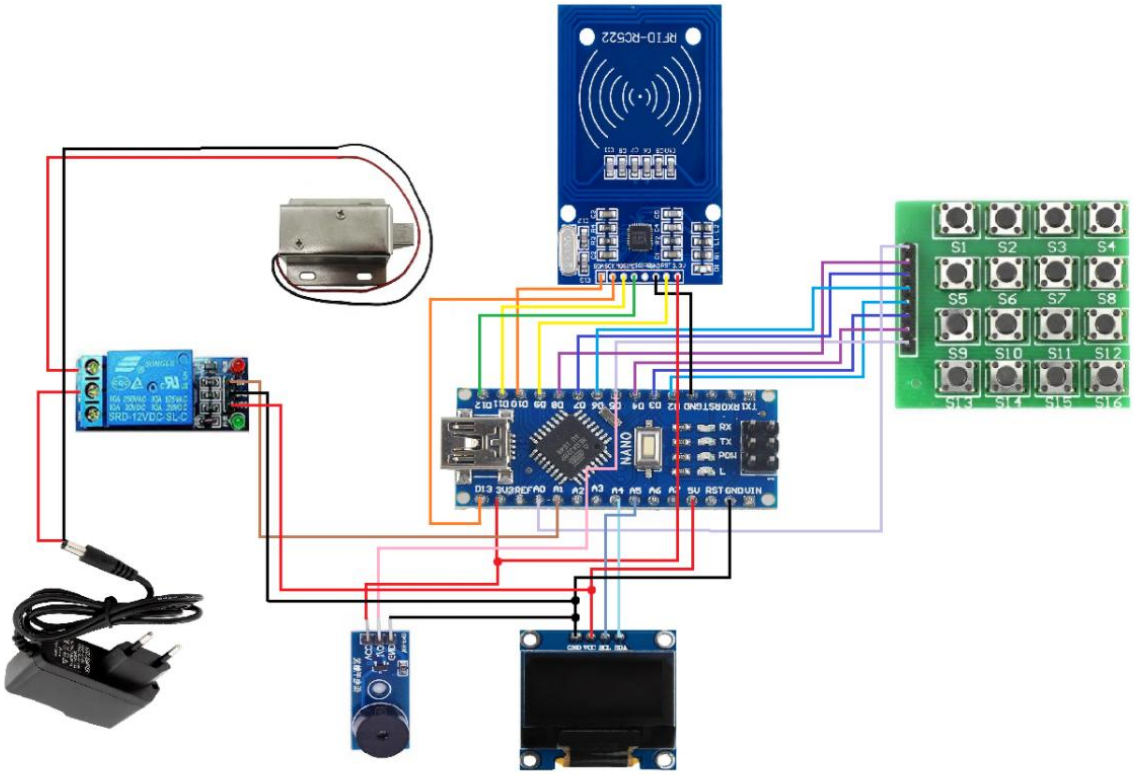

Рисунок 2.7 – Електрична схема підключення компонентів

Така схема підключення дозволяє мікроконтролеру Arduino ефективно взаємодіяти з усіма компонентами системи, зчитувати дані з пристроїв введення, обробляти їх та керувати виконавчими пристроями та пристроями виведення інформації, забезпечуючи функціонування розумного електронного замка відповідно до закладеної логіки.

3 РЕАЛІЗАЦІЯ РОЗУМНОГО ЕЛЕКТРОННОГО ЗАМКА

3.1 Програмна реалізація системи

Програмна частина є ядром розумного електронного замка, відповідаючи за збір та обробку даних, що надходять від користувача через пристрої введення, прийняття рішень щодо надання або відмови в доступі, а також керування виконавчими механізмами та індикацією стану системи. Вона побудована на базі мікроконтролера Arduino Nano і використовує низку стандартних та спеціалізованих бібліотек для взаємодії з апаратними компонентами. До ключових бібліотек, задіяних у програмному коді, належать: Wire.h для організації зв'язку за протоколом I2C, що використовується OLED-дисплеєм; SPI.h для забезпечення комунікації за інтерфейсом SPI, необхідним для роботи RFID-модуля MFRC522; MFRC522.h, що надає високорівневі функції для взаємодії з RFID-зчитувачем; Keypad.h для зручної обробки натискань на кнопках матричної клавіатури; а також бібліотеки Adafruit_SSD1306.h та Adafruit_GFX.h, які слугують для керування OLED-дисплеєм SSD1306 та виведення на нього текстової та графічної інформації. Кожна з цих бібліотек інкапсулює низькорівневу взаємодію з відповідним апаратним пристроєм, що дозволяє розробнику зосередитися на реалізації основної логіки роботи системи контролю доступу.

3.2 Ініціалізація компонентів

Функція setup() в середовищі Arduino виконується один раз на самому початку роботи програми – одразу після завантаження мікроконтролера або його перезапуску. Головне призначення цієї функції полягає у підготовці всіх апаратних компонентів системи до роботи та ініціалізації необхідних програмних змінних. Коректне виконання всіх операцій у setup() є критично

важливим для стабільної та передбачуваної роботи електронного замка.

Першим кроком є ініціалізація послідовного порту за допомогою команди Serial.begin(9600). Це дозволяє налаштувати зв’язок між мікроконтролером та комп’ютером на швидкості 9600 біт на секунду, що є стандартною практикою для виведення відлагоджувальної інформації та моніторингу стану системи під час розробки та тестування.

Далі відбувається налаштування цифрових та аналогових пінів мікроконтролера, які використовуються для керування периферійними пристроями. Пін A1, визначений у коді як RELAY_PIN, конфігурується як вихід (pinMode(RELAY_PIN, OUTPUT)) для керування релейним модулем, що комутує живлення електрозамка. Одразу після конфігурації на цей пін подається високий рівень сигналу (digitalWrite(RELAY_PIN, HIGH)). Враховуючи, що використаний релейний модуль активується низьким рівнем сигналу, як підтверджується логікою функції openLock, де для відкриття замка на пін реле подається низький рівень (LOW), встановлення високого рівня на старті гарантує, що реле буде деактивоване. Це означає, що електрозамок залишиться у закритому (заблокованому) стані при увімкненні живлення або перезавантаженні системи. Такий підхід є важливим аспектом безпеки, оскільки він запобігає ситуації, коли замок міг би випадково відкритися через збій живлення або програмний перезапуск, реалізуючи принцип fail-secure на рівні програмної ініціалізації. Аналогічно, пін D5, визначений як BUZZER_PIN, налаштовується як вихід (pinMode(BUZZER_PIN, OUTPUT)) для керування пасивним зумером. Для запобігання небажаним звукам при старті системи викликається спеціальна функція silenceBuzzer, яка гарантовано вимикає зумер.

Ініціалізація OLED-дисплея SSD1306 здійснюється за допомогою виклику display.begin(SSD1306_SWITCHCAPVCC, 0x3C). Тут 0x3C – це I2C адреса дисплея. Результат цієї операції перевіряється: якщо ініціалізація не вдалася (if(!display.begin(...))), у послідовний порт виводиться повідомлення про помилку (SSD1306 allocation failed), і виконання програми зупиняється у

нескінченному циклі (while(1)). Після успішної ініціалізації дисплей короткочасно активується (display.display()), очищується (display.clearDisplay()) і на нього виводиться привітальне повідомлення за допомогою функції showWelcomeScreen.

Ініціалізація RFID-зчитувача MFRC522, який використовує інтерфейс SPI для зв’язку з мікроконтролером, починається з активації самої шини SPI командою SPI.begin(). Після цього викликається функція mfrc522.PCD_Init() для ініціалізації модуля зчитувача. Для забезпечення чистого старту та уникнення можливих конфліктів з попередніми сеансами зв’язку з RFID-картами, виконуються команди mfrc522.PICC_HaltA() та mfrc522.PCD_StopCrypto1(). Перша команда переводить будь-яку активну карту в стан очікування, а друга деактивує криптографічні функції модуля, якщо вони були задіяні.

На завершення функції setup() ініціалізується змінна lastActivityTime = millis(), яка використовується для відстеження часу останньої взаємодії користувача з системою, що необхідно для реалізації функції таймауту неактивності.

У лістингу 3.1 наведено код функції setup() з відповідними коментарями, що пояснюють призначення кожного блоку інструкцій.

Лістинг 3.1 – Функція валідації isnan()

```
void setup() {
  Serial.begin(9600);
  pinMode(RELAY_PIN, OUTPUT);
  pinMode(BUZZER_PIN, OUTPUT);
)
  digitalWrite(RELAY_PIN, HIGH);
  silenceBuzzer();
  if(!display.begin(SSD1306_SWITCHCAPVCC, 0x3C)) {
    Serial.println("Не вдалося ініціалізувати SSD1306");
    while (1);
  }
  display.display();
  delay(2000);
  display.clearDisplay();
  showWelcomeScreen();
```

```
  SPI.begin(); // Ініціалізація SPI-шини
  mfrc522.PCD_Init();  // Ініціалізація зчитувача MFRC522
  Serial.println("RFID зчитувач ініціалізовано");
  lastActivityTime = millis();
  mfrc522.PICC_HaltA();
  mfrc522.PCD_StopCrypto1();
}
```

Процес ініціалізації забезпечує, що всі компоненти системи готові до виконання своїх функцій, а початковий стан системи є безпечним та визначеним.

3.3 Алгоритм роботи основного циклу

Функція loop() є центральною частиною будь-якої програми для Arduino, оскільки вона виконується безперервно в циклі після одноразового виконання функції setup(). Саме тут реалізована основна логіка роботи розумного електронного замка: опитування стану системи, обробка подій від RFID-зчитувача та клавіатури, взаємодія з користувачем через OLED-дисплей і зумер, а також прийняття рішень щодо керування замком.

На початку кожної ітерації перевіряється, чи встановлено прапорець systemLocked. Якщо він рівний true, система перебуває у режимі блокування. Тоді виконується оновлення екрану блокування викликом функції showLockedScreen(), яка виводить повідомлення про стан блокування та таймер зворотного відліку. Цей таймер оновлюється приблизно раз на секунду, щоб уникнути надмірного навантаження на дисплей. Якщо час блокування вичерпано (порівняння millis() - lockStartTime > LOCK_PERIOD), система розблоковується: прапорець systemLocked встановлюється в false, лічильник невдалих спроб failedAttempts обнуляється, виводиться вітальний екран викликом showWelcomeScreen(), а RFID-зчитувач повторно ініціалізується командою mfrc522.PCD_Init(). Після цього поточна ітерація циклу завершується оператором return, що забороняє виконання подальших дій, поки триває блокування.

Якщо система не заблокована, перевіряється таймаут неактивності. Якщо користувач почав взаємодію (прапорець welcomeScreenActive дорівнює false), але з моменту останньої активності минуло більше 7000 мілісекунд (millis() - lastActivityTime > TIMEOUT_PERIOD), виконується автоматичне повернення до початкового стану: викликається showWelcomeScreen(), прапорець welcomeScreenActive встановлюється в true, а індекс введених символів PIN-коду correctIndex скидається в 0.

Після цього виконується послідовне опитування пристроїв введення. За допомогою функції mfrc522.PICC_IsNewCardPresent() перевіряється наявність нової RFID-карти. Якщо карта виявлена і успішно зчитано її UID (mfrc522.PICC_ReadCardSerial()), UID формується у шістнадцятковий рядок та виводиться на OLED-дисплей. Якщо UID співпадає з авторизованим значенням (cardUID == allowedUID), лічильник failedAttempts скидається, викликається функція openLock() для відкриття замка, після чого через 7 секунд виконується closeLock() для закриття. Якщо UID не співпадає, лічильник збільшується, виконується denyAccess(), і якщо ліміт невдалих спроб перевищено, запускається блокування через виклик lockSystem().

Далі за допомогою функції keypad.getKey() перевіряється, чи було натиснуто клавішу на матричній клавіатурі. Якщо так, символ передається на перевірку функції checkKeyPress(key), яка відповідає за збір PIN-коду, його відображення у вигляді зірочок на екрані та порівняння з еталонним PIN-кодом. Якщо введено правильний PIN, лічильник невдалих спроб очищується, виконується відкриття замка і через 7 секунд – закриття.

Таким чином, цикл loop() забезпечує контроль за станом блокування, обробляє таймаут неактивності, опитує пристрої введення та виконує відповідні дії, забезпечуючи безпеку та зручність роботи системи.

3.4 Зчитування та перевірка RFID

Обробка ідентифікації за допомогою RFID-карти є одним із двох

основних способів автентифікації користувача в розробленій системі. Цей процес реалізований у межах основного циклу loop() та використовує функції бібліотеки MFRC522.h для взаємодії з модулем MFRC522.

Процес починається з перевірки наявності нової RFID-карти в зоні дії антени зчитувача. Функція mfrc522.PICC_IsNewCardPresent() повертає true, якщо виявлено карту, яка ще не була оброблена. Якщо така карта присутня, викликається функція mfrc522.PICC_ReadCardSerial(), яка намагається зчитати її унікальний ідентифікатор (UID). Якщо обидві ці функції успішно виконуються, система переходить до обробки отриманого UID.

При успішному виявленні та зчитуванні карти першочергово оновлюється час останньої активності (lastActivityTime = millis();) та встановлюється прапорець welcomeScreenActive = false;, сигналізуючи, що користувач взаємодіє з системою і вона вийшла з режиму очікування.

Далі відбувається формування UID карти у вигляді рядка. UID зчитується як масив байтів (mfrc522.uid.uidByte), де кожен байт представляє частину ідентифікатора. Ці байти послідовно перетворюються у шістнадцяткове представлення і конкатенуються в один рядок cardUID за допомогою циклу: for (byte i = 0; i < mfrc522.uid.size; i++) { cardUID += String(mfrc522.uid.uidByte[i], HEX); }. Після формування, отриманий рядок UID приводиться до верхнього регістру за допомогою методу cardUID.toUpperCase();. Це робиться для забезпечення коректного порівняння зі збереженим авторизованим UID (allowedUID), який також зберігається у верхньому регістрі, унеможливлюючи помилки через різний регістр символів (наприклад, 'a' та 'A').

Зчитаний та відформатований cardUID відображається на OLED-дисплеї з підписом Scanned UID: для інформації користувача. Це дозволяє користувачеві бачити, який UID був зчитаний, що може бути корисним для діагностики.

Ключовим етапом є перевірка доступу. Зчитаний cardUID порівнюється зі значенням змінної allowedUID, яка містить попередньо визначений UID

авторизованої карти (у даній реалізації це String allowedUID = "2380411C";).

Якщо зчитаний cardUID співпадає з allowedUID, лічильник невдалих спроб failedAttempts скидається на нуль, і викликається функція openLock() для активації реле та відкриття замка. Система очікує 7 секунд (delay(7000);), протягом яких замок залишається відкритим, після чого виконується закриття замка через виклик функції closeLock().

Якщо ж cardUID не співпадає з allowedUID, лічильник failedAttempts збільшується на одиницю, і викликається функція denyAccess(), яка інформує про відмову в доступі звуковим сигналом та повідомленням на дисплеї. Після цього перевіряється, чи не було перевищено максимально допустиму кількість невдалих спроб (if (failedAttempts >= MAX_FAILED_ATTEMPTS)). У разі досягнення ліміту активується функція lockSystem(), що тимчасово блокує систему.

Після завершення обробки карти, незалежно від результату перевірки UID, обов’язково викликаються функції mfrc522.PICC_HaltA() та mfrc522.PCD_StopCrypto1(). Перша переводить RFID-карту у неактивний стан (HALT), що відповідає вимогам протоколу ISO/IEC 14443-3 і дозволяє зчитувачу коректно виявляти цю ж карту при повторному піднесенні або іншу карту, яка з’явиться в полі дії. Відсутність цієї команди могла б призвести до постійної відповіді карти і блокування виявлення нових карт. Друга функція припиняє активні криптографічні операції на модулі MFRC522, що необхідно для підготовки до наступного циклу зчитування. Таке правильне використання цих команд забезпечує надійне і послідовне виявлення карт.

3.5 Обробка введення PIN-коду з клавіатури

Другим способом автентифікації в системі є введення персонального ідентифікаційного номера (PIN-коду) за допомогою матричної клавіатури 4x4. Обробка натискань клавіш реалізована у функції checkKeyPress(char key), яка викликається у головному циклі системи щоразу, коли відбувається

натискання клавіші.

При кожному натисканні клавіші оновлюється час останньої активності користувача і скидається прапорець вітального екрану, що сигналізує про активну взаємодію з системою. Функція збирає символи PIN-коду у статичний масив currentInput, що розрахований на 4 символи, і відстежує кількість введених символів за допомогою змінної correctIndex.

Під час введення користувачу на OLED-дисплеї відображається запит “Enter PIN:” і зірочки, які відповідають кількості введених символів, щоб приховати сам код. Після введення 4 символів відбувається порівняння введеного PIN-коду з еталонним, який заданий у коді. Якщо введений код співпадає, на екрані з’являється повідомлення про успішне введення, лічильник невдалих спроб скидається, а масив введення очищується для подальшої роботи. Функція повертає true, що в головному циклі призводить до відкриття замка.

У випадку, якщо введений PIN-код не відповідає еталонному, лічильник невдалих спроб збільшується, а на дисплеї відображається повідомлення про помилку з інформацією про кількість спроб, що залишилися. Масив введення очищується для повторного введення. Якщо кількість невдалих спроб досягає максимального ліміту, активується функція тимчасового блокування системи, яка захищає від атак методом перебору. Якщо система ще не заблокована, після короткої паузи користувачеві знову пропонується ввести PIN-код. Функція в такому випадку повертає false.

Важливо, що стан введення PIN-коду скидається не тільки після повної спроби, але й при таймауті неактивності, що забезпечує початок кожної нової спроби з чистого стану. Це підвищує безпеку та надійність системи, запобігаючи накопиченню частково введених даних, які можуть призвести до помилок або вразливостей. У лістингу 3.2 наведено функцію checkKeyPress(char key), яка ілюструє описану логіку.

Лістинг 3.2 — Функція обробки введення PIN-коду

```
bool checkKeyPress(char key) {
  static char currentInput[4] = {0};
  if (correctIndex < 4) {
    currentInput[correctIndex] = key;
    correctIndex++;
    display.clearDisplay();
    display.setCursor(0, 0);
    display.setTextSize(1);
    display.println("Enter PIN:");
    display.setCursor(0, 15);
    for (int i = 0; i < correctIndex; i++) {
      display.print("*");
    }
    display.display();
    if (correctIndex == 4) {
      bool correctPin = true;
      for (int i = 0; i < 4; i++) {
        if (currentInput[i] != correctSequence[i]) {
          correctPin = false;
          break;
        }
      }
      correctIndex = 0;
      if (correctPin) {
        display.clearDisplay();
        display.setTextSize(1);
        display.setCursor(0, 0);
        display.println("PIN Accepted");
        display.display();
        for (int i = 0; i < 4; i++) {
          currentInput[i] = 0;
        }
        return true;
      } else {
        failedAttempts++;

        display.clearDisplay();
        display.setTextSize(1);
        display.setCursor(0, 0);
        display.println("PIN Incorrect");

        if (failedAttempts < MAX_FAILED_ATTEMPTS) {
          display.setCursor(0, 20);
          display.print("Attempts left: ");
          display.print(MAX_FAILED_ATTEMPTS - failedAttempts);
        }
        display.display();
        for (int i = 0; i < 4; i++) {
          currentInput[i] = 0;
        }
```

```
        if (failedAttempts >= MAX_FAILED_ATTEMPTS) {
          lockSystem();
          return false;
        }

        delay(1000);
        display.clearDisplay();
        display.setCursor(0, 0);
        display.setTextSize(1);
        display.println("Enter PIN:");
        display.display();
      }
    }
  }
  return false;
    }
```

3.6 Керування замком та зумером

Керування фізичним станом електронного замка здійснюється через релейний модуль, а звуковий зворотний зв’язок користувачу забезпечує пасивний зумер. Для цього у програмі реалізовані окремі функції — openLock(), closeLock(), denyAccess() та silenceBuzzer().

Функція openLock() виконує відкриття замка після успішної автентифікації. Вона подає низький рівень сигналу на пін реле RELAY_PIN за допомогою команди digitalWrite(RELAY_PIN, LOW);. Через особливості підключення реле саме низький рівень активує його, замкнувши ланцюг живлення замка і викликаючи його відкриття. Одночасно активується короткий звуковий сигнал із частотою 600 Гц тривалістю 300 мілісекунд, який генерується командою tone(BUZZER_PIN, 600); з наступним викликом silenceBuzzer() для припинення звуку. На OLED-дисплеї відображається повідомлення Lock Opened, яке інформує користувача про успішне відкриття (лістинг 3.3).

Лістинг 3.3 – Функція openLock()

```
void openLock() {
```

```
  digitalWrite(RELAY_PIN, LOW); // Включення реле (низький рівень)
  tone(BUZZER_PIN, 600);        // Сигнал 600 Гц
  delay(300);
  silenceBuzzer();              // Вимкнення зумера
  display.clearDisplay();
  display.setCursor(0, 0);
  display.setTextSize(1);
  display.println("Lock Opened");
  display.display();
}
```

Після виклику openLock() в основному циклі передбачена затримка 7000 мілісекунд, протягом яких замок залишається відкритим. Це забезпечує автоматичне закриття замка без участі користувача, що є важливою мірою безпеки і запобігає випадковому залишенню замка відкритим.

Закриття замка відбувається у функції closeLock(). Вона встановлює високий рівень сигналу на пін реле (digitalWrite(RELAY_PIN, HIGH);), що деактивує реле і вимикає живлення замка, закриваючи його. Одночасно з цим генерується звуковий сигнал із частотою 500 Гц тривалістю 300 мілісекунд, що відрізняється від сигналу відкриття і дозволяє користувачеві розрізняти ці дії за звуком. На дисплеї виводиться повідомлення Lock Closed. Для підтримки коректної роботи системи скидається стан введення PIN-коду (correctIndex = 0), оновлюється час останньої активності, а після затримки в 1 секунду на дисплеї знову відображається вітальний екран (лістинг 3.4).

Лістинг 3.4 – Функція closeLock()

```
void closeLock() {
  digitalWrite(RELAY_PIN,  HIGH);  //  Вимикання  реле  (високий
рівень)
  tone(BUZZER_PIN, 500);          // Сигнал 500 Гц
  delay(300);
  silenceBuzzer();
  display.clearDisplay();
  display.setCursor(0, 0);
  display.setTextSize(1);
  display.println("Lock Closed");
  display.display();
  correctIndex = 0;
  lastActivityTime = millis();
```

```
  delay(1000);
  showWelcomeScreen();
  welcomeScreenActive = true;
}
```

У разі невдалої спроби доступу, наприклад, при введенні невірного PIN-коду або зчитуванні неавторизованої RFID-карти, виконується функція denyAccess(). Вона генерує більш тривалий і високочастотний звуковий сигнал (800 Гц, 500 мс), який чітко сигналізує про помилку. На дисплеї виводиться повідомлення Access Denied, а якщо ліміт спроб ще не перевищено, додатково показується кількість спроб, що залишилися (лістинг 3.5).

Лістинг 3.5 – Функція denyAccess()

```
void denyAccess() {
  tone(BUZZER_PIN, 800);         // Сигнал 800 Гц
  delay(500);
  silenceBuzzer();
  display.clearDisplay();
  display.setCursor(0, 0);
  display.setTextSize(1);
  display.println("Access Denied");
  if (failedAttempts < MAX_FAILED_ATTEMPTS) {
    display.setCursor(0, 20);
    display.print("Attempts left: ");
    display.print(MAX_FAILED_ATTEMPTS - failedAttempts);
  }
  display.display();
}
```

Функція silenceBuzzer() відповідає за повне припинення звукового сигналу зумера. Вона виконує команду noTone(BUZZER_PIN); для вимкнення тону, встановлює низький рівень на відповідному піні та переводить його у режим виходу. Це дозволяє уникнути залишкових звуків або випадкових спрацьовувань зумера (лістинг 3.6).

Лістинг 3.6 – Функція silenceBuzzer()

```
void silenceBuzzer() {
  noTone(BUZZER_PIN);          // Зупинка генерації тону
  digitalWrite(BUZZER_PIN, LOW);
  pinMode(BUZZER_PIN, OUTPUT); // Гарантія встановлення піна у
вихідний режим
}
```

Загалом поєднання чітких звукових сигналів різної тональності та тривалості з текстовими повідомленнями на OLED-дисплеї підвищує зручність користування пристроєм. Користувач швидко розпізнає результати своїх дій і поточний стан замка, що значно покращує інтуїтивність і безпеку системи.

3.7 OLED SSD1306

OLED-дисплей на базі контролера SSD1306 є основним засобом візуальної взаємодії між електронним замком і користувачем. Він інформує про поточний стан системи, відображає підказки та результати виконаних операцій. Для керування дисплеєм використовуються функції з бібліотек Adafruit_SSD1306.h та Adafruit_GFX.h, які дозволяють виводити текст, задавати його розмір, колір (білий на чорному тлі) та позицію курсора.

Основний екран привітання реалізує функція showWelcomeScreen(). Вона очищає дисплей командою display.clearDisplay(), встановлює розмір тексту за допомогою display.setTextSize(1) і колір тексту display.setTextColor(SSD1306_WHITE). Після цього на екрані з’являється назва системи Smart Lock і підказка Waiting for input..., що сигналізує про готовність до взаємодії.

Якщо система переходить у стан тимчасового блокування через перевищення ліміту невдалих спроб доступу, викликається функція showLockedScreen(). Вона обчислює час, що залишився до розблокування, віднімаючи від загального періоду блокування (LOCK_PERIOD) час, який минув від початку блокування (millis() - lockStartTime). Отриманий час у

мілісекундах переводиться в секунди для зручності відображення. На дисплеї виводиться повідомлення SYSTEM LOCKED та зворотний відлік часу у секундах, який оновлюється приблизно раз на секунду (лістинг 3.7).

Лістинг 3.7 – showLockedScreen()

```
void showLockedScreen() {
  unsigned long timeElapsed = millis() - lockStartTime;
  unsigned long timeRemaining = LOCK_PERIOD - timeElapsed;
  int secondsRemaining = timeRemaining / 1000;
  display.clearDisplay();
  display.setTextSize(1);
  display.setTextColor(SSD1306_WHITE);
  display.setCursor(0, 0);
  display.println("SYSTEM LOCKED");// Повідомлення про блокування
  display.setCursor(0, 20);
  display.print("Wait: ");
  display.print(secondsRemaining);    //    Відображення    часу
очікування
  display.println(" sec");
  display.display();
     }
```

Крім цих основних даних, дисплей використовується для відображення додаткової інформації у різних ситуаціях. Під час зчитування RFID-карти на екрані показується текст Scanned UID шрифтом меншого розміру (display.setTextSize(1)), а нижче більшим шрифтом (display.setTextSize(2)) – унікальний ідентифікатор карти (UID), що допомагає користувачу побачити, який саме UID було зчитано.

Під час введення PIN-коду дисплей виводить текст Enter PIN, а під ним послідовність символів * кількість яких відповідає кількості введених цифр. Це дає візуальний зворотний зв’язок, не розкриваючи сам PIN.

Після перевірки PIN-коду або RFID-карти на дисплеї з’являються відповідні статусні повідомлення – PIN Accepted, PIN Incorrect, Access Denied, Lock Opened, Lock Closed. Якщо система не заблокована, у разі невірного введення також виводиться інформація про кількість залишкових спроб.

Для виведення тексту використовуються стандартні методи бібліотеки

Adafruit GFX: очищення буфера (display.clearDisplay()), позиціювання курсора (display.setCursor(x, y)), зміна розміру та кольору шрифту (display.setTextSize(), display.setTextColor()), а також виведення тексту (display.print(), display.println()). Виведення завершується командою display.display(), яка передає сформоване зображення на OLED-екран.

Завдяки такій організації інформаційних екранів OLED-дисплей забезпечує зрозумілу та інформативну взаємодію користувача із системою, підвищуючи зручність та безпеку експлуатації електронного замка.

4 ТЕСТУВАННЯ СИСТЕМИ

Для тестування системи розумного замку було зібрано прилад згідно електричної схеми пристрою що була розроблена у процесі планування. Після збирання системи на мікроконтролер плати Arduino Nano було завантажено розроблене програмне забезпечення. Пристрій у робочому стані показаний на рисунку 4.1.

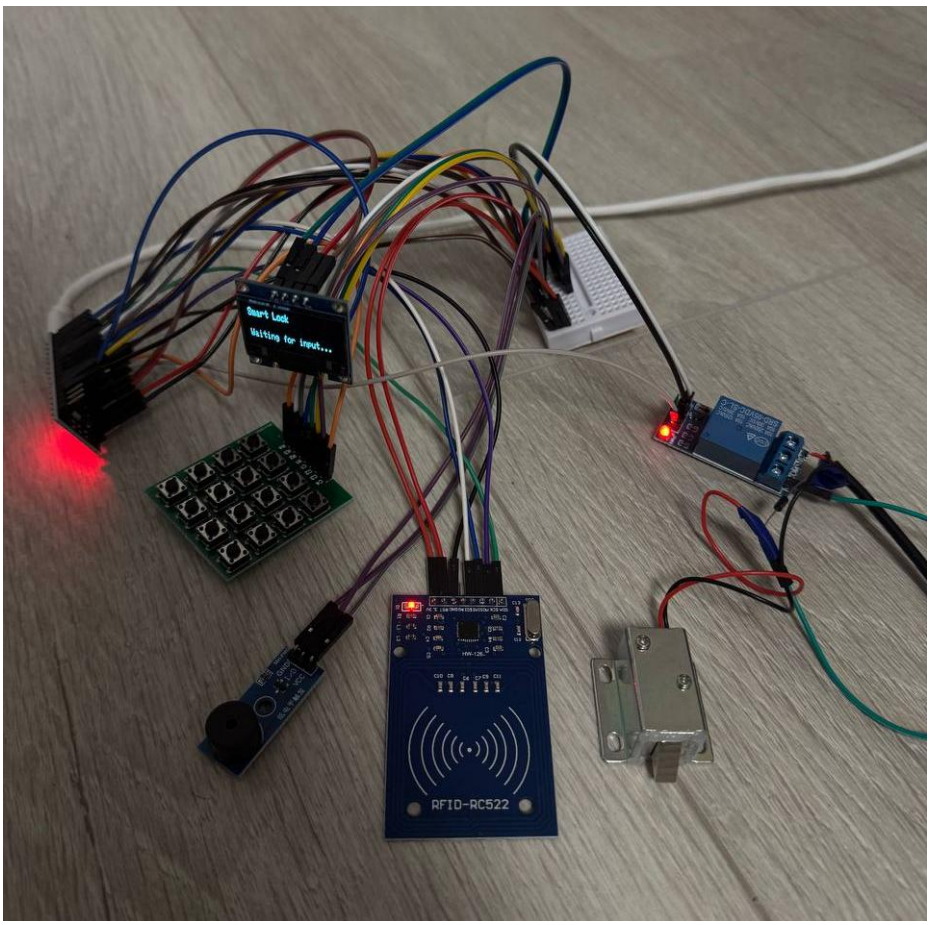

Рисунок 4.1 – Зібрана система розумного замку

На цьому етапі тестування було підтверджено правильну роботу ініціалізації RFID-зчитувача та клавіатури. Реле залишалося вимкненим при старті, що гарантувало закритий стан замка. Зумер при включенні не видавав звуків.

При піднесенні авторизованої RFID-карти з UID 2380411C на дисплеї відображався UID та повідомлення Lock Opened (рисунок 4.2). Замок відчинявся на 7 секунд, супроводжуючись звуковим сигналом. Після цього замок закривався зі звуковим підтвердженням.

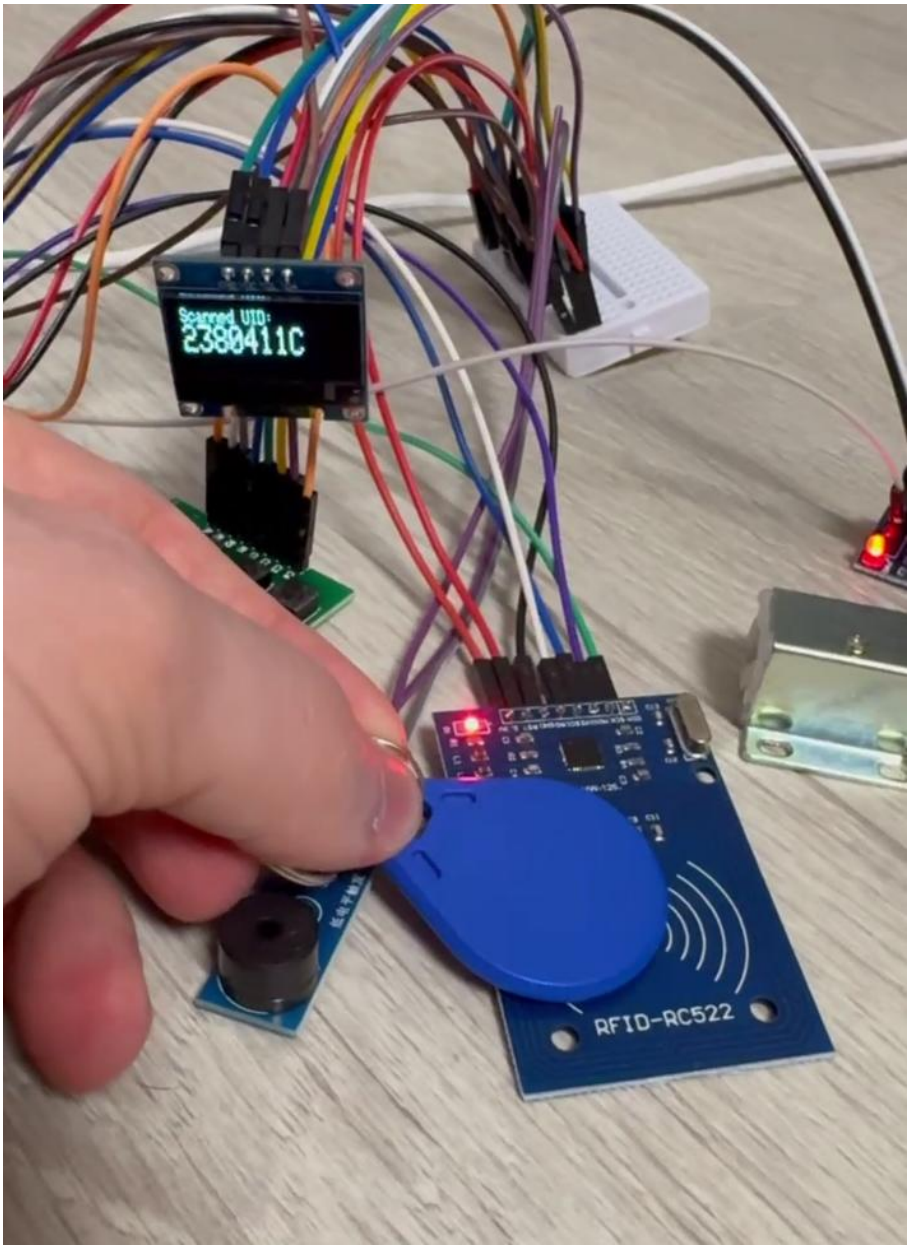

Рисунок 4.2 – Зчитування RFID мітки

Якщо підносити неавторизовану карту, система показувала UID і повідомлення Access Denied, зумер видавав сигнал відмови, а замок залишався

зачиненим (рисунок 4.3).

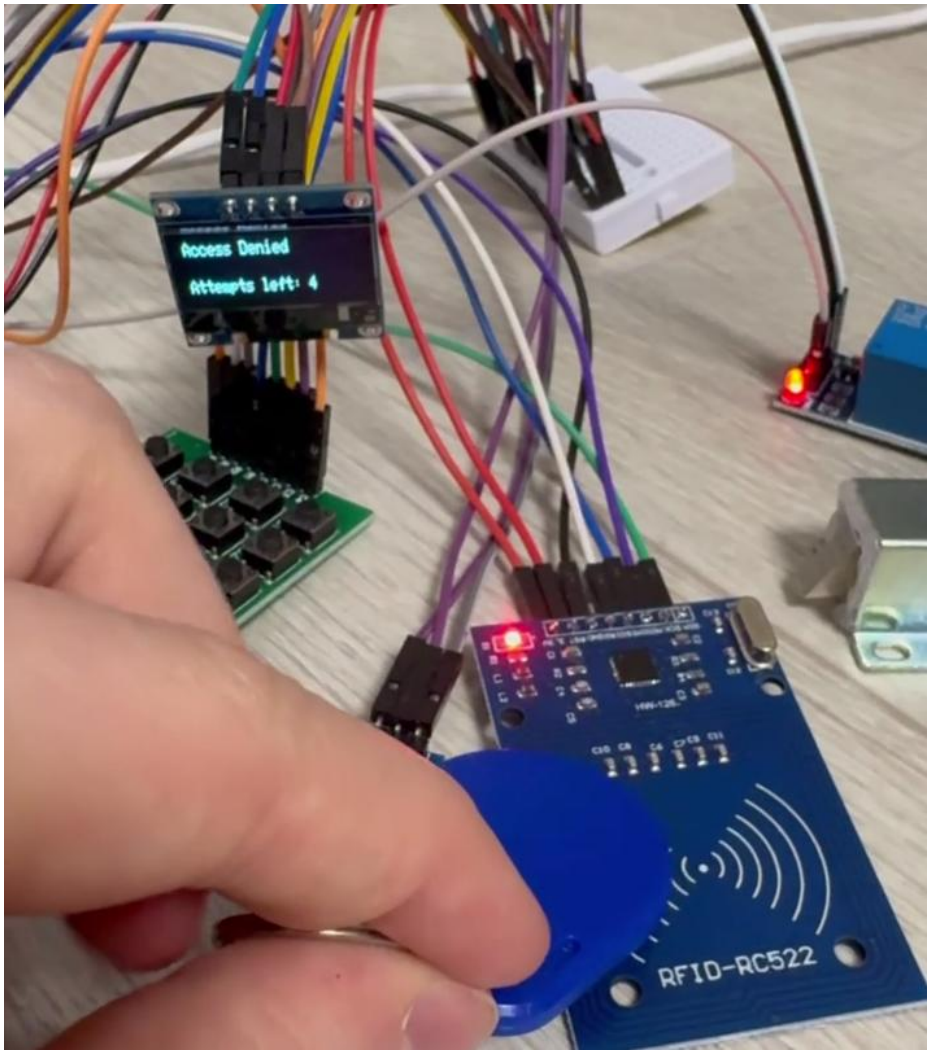

Рисунок 4.3 – Використання неавторизованої RFID мітки

Введення PIN-коду теж тестувалося на правильність і неправильність. При введенні правильного PIN 4842 на дисплеї з’являлися зірочки замість цифр, після чого замок відчинявся, а при неправильному PIN відображалися повідомлення про помилку та відмову в доступі (рисунок 4.4).

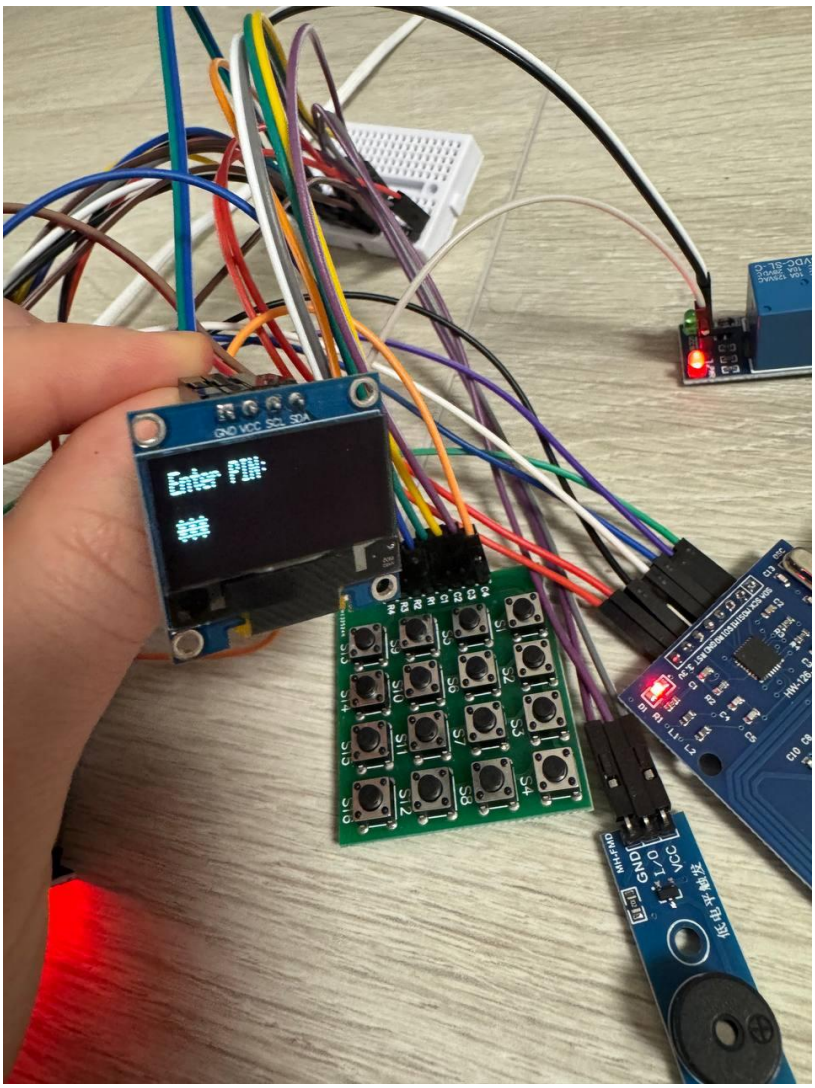

Рисунок 4.4 – Введення пінкоду

Після п’яти неправильних спроб отримання доступу система блокувалася на 60 секунд, показуючи повідомлення SYSTEM LOCKED і таймер. Зумер видавав потрійний сигнал блокування. Після завершення блокування система автоматично поверталася до роботи.

ВИСНОВКИ

У ході виконання кваліфікаційної роботи було розроблено та реалізовано розумний електронний замок на базі мікроконтролера Arduino Nano з використанням RFID-зчитувача MFRC522, матричної клавіатури 4x4, OLED-дисплея та релейного модуля для керування електромеханічним замком. Проаналізовано існуючі методи контролю доступу та обрано оптимальні апаратні компоненти для реалізації доступної і надійної системи. Програмне забезпечення, написане мовою C++ у середовищі Arduino IDE, забезпечує коректну взаємодію між апаратурою та логіку автентифікації користувача за допомогою PIN-коду або RFID-картки.

Розроблена система підтримує надійний захист від несанкціонованого доступу, включаючи механізм тимчасового блокування після кількох невдалих спроб. OLED-дисплей відображає інформативні повідомлення, які допомагають користувачу орієнтуватися у стані системи, а зумер забезпечує звуковий зворотний зв’язок для підтвердження дій. Тестування прототипу показало стабільність роботи, своєчасність відгуку на введення даних та коректність виконання функцій відкриття і закриття замка, а також адекватну обробку некоректних спроб доступу.

Реалізований пристрій є доступним та функціональним рішенням для контролю доступу, що може бути використано у побутових або офісних умовах для підвищення рівня безпеки. Подальший розвиток проекту може включати інтеграцію з мережевими системами контролю доступу, розширення функціоналу біометричними методами і вдосконалення апаратної платформи для підвищення надійності та зручності користування.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. SailPoint. What Are the Different Types of Access Control Systems. URL: https://www.sailpoint.com/identity-library/what-are-the-different-types-of-access-control-systems (дата звернення: 16.05.2025).

2. Palo Alto Networks. Access Control. URL: https://www.paloaltonetworks.com/cyberpedia/access-control (дата звернення: 18.05.2025).

3. Solis. Класифікація систем контролю доступу. URL: http://solis.in.ua/klasyfikatsiya-system-kontrolyu-dostupu.html (дата звернення: 16.05.2025).

4. BAS-IP. Що таке СКУД і як організувати систему контролю доступу в офісі чи іншому об'єкті. URL: https://bas-ip.kiev.ua/shcho-take-skud-i-yak-orhanizuvaty-systemu-kontroliu-dostupu-v-ofisi-chy-inshomu-obiekti/ (дата звернення: 16.05.2025).

5. Zamochniki. Як працює електрозамок і де він застосовується. URL: https://zamochniki.com.ua/yak-pratsiuie-elektrozamok-i-de-vin-zastosovuietsya (дата звернення: 16.05.2025).

6. Perfect Okna. Магнітний та електронний замок на захисті вхідних дверей. URL: https://perfect-okna.com.ua/statti/magnitnii-ta-elektronnii-zamok-na-zahisti-vhidnih-dverei/ (дата звернення: 16.05.2025).

7. NXP. MFRC522 Data Sheet. URL: https://www.nxp.com/docs/en/data-sheet/MFRC522.pdf (дата звернення: 16.05.2025).

8. NiceOne. 4x4 Matrix Membrane Keypad: A Comprehensive Guide. URL: https://www.niceone-keypad.com/4x4-matrix-membrane-keypad-a-comprehensive-guide/ (дата звернення: 16.05.2025).

9. Mouser Electronics. Soldered 333099. URL: https://www.mouser.com/datasheet/2/1398/Soldered_333099-3395096.pdf (дата

звернення: 18.05.2025).

10. Steemit. Arduino 101: Controlling a Solenoid Lock. URL: https://steemit.com/utopian-io/@ted7/arduino-101-controlling-a-solenoid-lock (дата звернення: 17.05.2025).