Міністерство освіти і науки України

Луцький національний технічний університет
(повне найменування закладу вищої освіти)

Факультет комп'ютерних та інформаційних технологій
(повне найменування факультету)

Кафедра комп'ютерної інженерії та кібербезпеки
(повне найменування кафедри)

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

АВТОНОМНИЙ ПРИСТРІЙ ДЛЯ ВИЗНАЧЕННЯ РОБОЧОЇ ЧАСТОТИ ТА ДУБЛЮВАННЯ RFID-МІТОК

AUTONOMOUS DEVICE FOR DETERMINING THE OPERATING FREQUENCY AND DUPLICATING RFID LABELS

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

освітня програма Комп'ютерна інженерія
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи КІс-21
Іщук Владислав Вікторович

(підпис)

Керівник:
к.т.н., доцент
Костючко Сергій Миколайович

(підпис)

Кваліфікаційну роботу
допущено до захисту
« _____ » червня 2023 р.
Гарант освітньої програми:
к.т.н., доцент
Лавренчук Світлана Василівна

(підпис)

Луцьк – 2023 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *комп'ютерної інженерії та кібербезпеки*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *123 Комп'ютерна інженерія*
Освітня програма: *«Комп'ютерна інженерія»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ проф. Н.Чернящук
«_____» _____ 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Іщуку Владиславу Вікторовичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи *Автономний пристрій для визначення робочої частоти та дублювання RFID-міток*

Керівник роботи *к.т.н., доцент Костючко С.М.*

затверджені наказом закладу вищої освіти від «28» грудня 2022 року № 982/01-02
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи 01.06.2023р.

3. Вихідні дані до роботи *Джерелом розробки є науково-технічна література та публікації в періодичних виданнях з даного питання, опубліковані зарубіжні та вітчизняні роботи в даній області та різні інтернет-ресурси технічного спрямування*

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити):
Вступ
Аналітичний огляд предметної області
Теоретичні аспекти та вибір компонентів
Практична реалізація проєкту
Висновки

5. Перелік графічного (ілюстративного) матеріалу:

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналітичний огляд предметної області	*Костючко С.М.*		
Теоретичні аспекти та вибір компонентів	*Костючко С.М.*		
Практична реалізація проєкту	*Костючко С.М.*		
Висновки	*Костючко С.М.*		

7. Дата видачі завдання 01.11.2022 р.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	*Обгрунтування теми*	До 15.11.2022 р.	Виконано
2.	*Огляд літератури із досліджуваної проблеми*	До 17.12.2022 р.	Виконано
3.	*Розробка технічного завдання, вибір методів та засобів реалізації задачі*	До 02.02.2023 р.	Виконано
4.	*Розробка структури прототипу та проектування системи*	До 02.03.2023 р.	Виконано
5.	*Описати програмне та апаратне середовище функціонування об'єкта програмування*	До 02.04.2023 р.	Виконано
6.	*Практична реалізація об'єкта програмування*	До 15.04.2023 р.	Виконано
7.	*Висновки та пропозиції*	До 02.05.2023 р.	Виконано
8.	*Формування додатків*	До 15.05.2023 р.	Виконано
9.	*Нормоконтроль*	До 25.05.2023 р.	Виконано
10.	*Інструментальна перевірка на академічний плагіат*	До 01.6.2023 р	Виконано
11.	*Представлення кваліфікаційної роботи бакалавра до захисту*	До 07.06.2023 р.	Виконано

Здобувач вищої освіти _____ Іщук В.В.
(підпис) (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ Костючко С.М.
(підпис) (прізвище, ініціали)

АНОТАЦІЯ

Іщук В.В. Автономний пристрій для визначення робочої частоти та дублювання RFID-міток. Рукопис.

Кваліфікаційна робота бакалавра ОП «Комп'ютерна інженерія» спеціальності 123 Комп'ютерна інженерія. Луцький національний технічний університет. Луцьк, 2023.

Кваліфікаційна робота складається з вступу, трьох розділів, висновків, списку використаних джерел, додатку.

Перший розділ присвячено огляду літературних джерел, проведено аналіз видів ключів, технологій ключів, домофонів

В другому розділі здійснено дослідження існуючих рішень для зчитування та копіювання Touch-memory та RFID міток, виділено основні переваги та недоліки. Досліджено усі складові пристрою. Досліджено усі нюанси під час створення копіювальника.

В третьому розділі розроблено схему приладу, проведено аналіз ефективності роботи, розроблено функціонал та інтерфейс, виконано тестування та внесено необхідні виправлення.

Результатом є готовий автономний пристрій для копіювання домофонних ключів, який може бути використаний як для звичайного користувача домофоном, так і для спецслужб.

Об'єкт дослідження – засоби та форми визначення робочої частоти та дублювання безконтактних міток.

Предмет розробки – автономний пристрій для визначення робочої частоти та дублювання rfid-міток.

Метою роботи є вибір комплектуючих для створення дублікатору контактних домофонних ключів на ArduinoNano. Покроковий опис монтажу, програмування та опис компонентів.

Ключові слова: Arduino Nano, Arduino IDE, RFID memory, зчитування і запис, дублікатор домофонних ключів, пайка, ключі.

ABSTRACT

Ishchuk V.V. Autonomous device for determining the operating frequency and duplicating RFID tags. Manuscript.

Bachelor's qualifying thesis of the OP «Computer Engineering» specialty 123 Computer Engineering. Lutsk National Technical University. Lutsk, 2023.

The qualification work consists of an introduction, three sections, conclusions, a list of used sources, and an appendix.

The first chapter is devoted to the review of literary sources, an analysis of types of keys, key technologies, and intercoms was carried out.

In the second section, a study of existing solutions for reading and copying Touch-memory and RFID tags is carried out, the main advantages and disadvantages are highlighted. All components of the device have been studied. All the nuances during the creation of the copier have been studied.

In the third section, the scheme of the device was developed, the performance analysis was carried out, the functionality and interface were developed, testing was carried out and the necessary corrections were made.

The result is a ready-made stand-alone device for copying intercom keys, which can be used both for an ordinary intercom user and for special services.

The object of research is the means and forms of determining the working frequency and duplication of contactless tags.

The subject of development is an autonomous device for determining the operating frequency and duplicating rfid tags.

The purpose of the work is the selection of components for creating a duplicator of contact intercom keys on ArduinoNano. Step-by-step description of installation, programming and description of components.

Keywords: Arduino Nano, Arduino IDE, RFID memory, reading and writing, intercom key duplicator, soldering, keys.

ЗМІСТ

ВСТУП

Актуальність теми. В наші дні технології стрімко розвиваються, і ключі також не є винятком. Завдяки прогресу технологій, ця галузь досягла непобачених величин, і ключі стали невід'ємною частиною нашого повсякденного життя. Сьогодні існує безліч видів ключів, починаючи зі звичайних квартирних і автомобільних, навіть наші пальці можуть виступати в ролі ключів. Звичайний пароль від електронної пошти також виступає у ролі ключа.

Застосування такого пристрою може сприяти швидкому та точному зчитуванню RFID-міток, а також уможливити дублювання міток для забезпечення надійності та резервування даних. Враховуючи зростаючу популярність технології RFID та її широке використання у різних галузях, такий пристрій може мати значний потенціал в промисловості, логістиці, безпеці та інших сферах, де потрібна швидка та надійна ідентифікація об'єктів.

Метою роботи є вибір комплектуючих для створення дублікатору контактних домофонних ключів на ArduinoNano. Покроковий опис монтажу, програмування та опис компонентів.

Об'єкт дослідження – засоби та форми визначення робочої частоти та дублювання безконтактних міток.

Предмет розробки – автономний пристрій для визначення робочої частоти та дублювання rfid-міток.

Завдання, які необхідно виконати:

– здійснити аналіз видів ключів, технологій ключів, домофонів;

– провести дослідження існуючих рішень для зчитування та копіювання Touch-memory та RFID міток;

– здійснити розробку схеми приладу;

– розробити функціонал та інтерфейс, виконати тестування та внести необхідні виправлення.

РОЗДІЛ 1
АНАЛІТИЧНИЙ ОГЛЯД ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Ключі та домофони

«Ключ» – це термін з великою кількістю значень, і відповісти однозначно на питання «Що таке ключ?» досить складно. У сучасному світі існує безліч видів ключів (рисунок 1.1):

- Гайковий ключ: інструмент, який використовується для загвинчування або відгвинчування гайок, болтів, труб, штанг.
- Музичний ключ: знак у музичній нотації, який розташовує ноти на нотному стані.
- Ключ для натягування струн: інструмент, що використовується для натягування струн на музичних інструментах.
- Ключ шифрування: інформація, використовувана для шифрування або дешифрування повідомлень.
- Ключ для замикання та відмикання замків: інструмент для замикання та розблокування замків, засувів і т.д. Сюди входять як звичайні квартирні ключі, так і автомобільні ключі, які можуть містити імобілайзери.

Рисунок 1.1 – Види квартирних ключів [11]

Починаючи з 1995 року, на автомобілях почали встановлювати імобілайзери, які суттєво підвищили безпеку транспортних засобів від крадіжок

[4]. Імобілайзер – це пристрій, який необхідно мати для запуску двигуна автомобіля (рисунок 1.2). Він використовує технологію радіочастотного розпізнавання електронного ключа (транспондера, «мітки», картки), який повинен бути у водія автомобіля.

Рисунок 1.2 – Іммобілайзер [4]

Процес розпізнавання розпочинається з опитування радіоприймальним пристроєм (читачем) транспондера, розташованого у транспортному засобі. Опитування полягає в випромінюванні коротким радіоімпульсом, який передає енергію до антени транспондера і заряджає його накопичувальний конденсатор. Цей конденсатор виконує роль джерела живлення. Після закінчення радіоімпульсу, транспондер випромінює кодований сигнал, який є ідентифікатором власника.

Зазвичай фізичне розташування транспондера знаходиться у пластиковій кришці автомобільного ключа. Цей тип ключа також називають «ключ з електронним чипом».

У деяких автомобілях відсутня механічна складова ключа, і транспондер розташовується у самостійному корпусі, який зазвичай виготовлений з пластмаси. Цей корпус, який можна носити, наприклад, у кишені, називається брелоком. Для запуску двигуна автомобіля у такому випадку використовується кнопка або тумблер, якщо брелок знаходиться всередині автомобіля. Такий

брелок також називається «смарт-ключем» (рисунок 1.3). Система безключового доступу працює за наступним алгоритмом: коли власник наближається до автомобіля на певну відстань, електроніка розпізнає код смарт-ключа, який знаходиться в кишені або гаманці, після чого вимикає захист і розблоковує замки дверей. Після сідання водія у салон він натискає кнопку запуску, і двигун заводиться.

Для запобігання несанкціонованому доступу використовується спеціальний код, який генерується за допомогою особливого алгоритму шифрування. Цей код передається на певній частоті. Важливо, щоб автомобіль отримував код від смарт-ключа у режимі реального часу (лічильник ведеться в секундах), тому спроби відкрити чужий автомобіль безуспішні. Звісно, теоретично можливо викрасти будь-який автомобіль, включаючи ті, що мають таку систему доступу. Але для зловмисників потрібне дуже дороге обладнання і певний рівень досвіду в схожих справах.

Рисунок 1.3 – «Смарт ключі» [1]

Порівняно зі звичайним механічним ключем з імобілайзером, смарт-ключ є набагато безпечнішим, але водночас він коштує більше і є складнішим з електронної точки зору. «Смарт-ключ» [1], [3] працює за принципом, схожим на домофонний ключ, який також передає цифровий код до приймача, що надає доступ до функцій. Тож давайте перейдемо до обговорення домофонів.

Домофон є зручним і доступним способом відкривати двері та спілкуватися з відвідувачами (рисунок 1.4). Уявити звичайний житловий будинок без домофона важко. Основною перевагою домофонів є забезпечення безпеки для житлових і робочих приміщень. Основні складові частини домофонів :

- Переговорний пристрій аудіодомофона використовується для спілкування між відвідувачем та особами всередині будівлі.
- Панель виклику розташовується біля входу або на дверях приміщення і дозволяє відвідувачу зв'язатися з людьми всередині. Вона може бути антивандальною і містити різні функціональні елементи, такі як кнопки цифрового набору, підсвічування клавіатури, приховану відеокамеру спостереження, зчитувач контактного ключа або смарт-карти.
- Комутатор [2] використовується для перемикання сигналу до різних квартир. Він забезпечує перенаправлення сигналу до відповідного переговорного пристрою, залежно від набраного номера. Комутатор зазвичай встановлюється в багатоквартирних домофонах і може бути вбудованим у панель виклику. Він працює на засадах телефонної системи. За допомогою комутатора відвідувач може представитися і повідомити про мету свого візиту.
- Абонентський пристрій є пристроєм, за допомогою якого люди всередині можуть запитати відвідувача про його ідентифікацію та мету візиту, особливо якщо вони незнайомі. Цей пристрій зазвичай має кнопку відкриття, яка дозволяє відчинити замок дверей, якщо на дверях встановлений електричний замок. Також абонентський пристрій може мати екран, який підключений до камери спостереження, що дозволяє переглянути зображення відвідувача.
- Замикаючий пристрій використовується в сучасних домофонах для керування електричним замком, як хоча замок не є частиною самого домофона, а лише виконує команду на відкриття дверей. Існують різні варіанти замків:

1. Електромеханічний замок – це засув, який висувається за допомогою електромагніту або електродвигуна.

2. Електромагнітний замок – це електромагніт, який утримує двері закритими. У разі відключення живлення замок автоматично переходить у положення «відкрито».

Рисунок 1.4 – Домофон [5]

Дійсно, дротові та бездротові домофони [5], [6] мають свої переваги і недоліки. Ось деякі переваги дротових домофонів:

- Висока безпека: Дротові домофони забезпечують високий рівень безпеки, оскільки передача даних відбувається по проводам. Це унеможливлює перехоплення або заміну сигналу, забезпечуючи надійну захист від зламу.
- Дальність дії: Завдяки проводовому з'єднанню між внутрішнім і зовнішнім блоками, дротові домофони можуть мати більшу дальність дії. Внутрішній блок можна розмістити на значній відстані від зовнішнього блока, що дозволяє зручно організувати систему зв'язку навіть у великих будинках або спорудах.
- Вартість: Дротові домофони зазвичай є більш доступними з точки зору вартості. Це пов'язано з їх простішою конструкцією, оскільки не потрібно використовувати бездротові технології передачі сигналу.

Проте, слід також зазначити, що дротові домофони мають свої недоліки, такі як обмежена мобільність через необхідність проводів та складність установки через потребу прокладання проводів. В цьому контексті бездротові домофони можуть бути більш зручними та гнучкими у використанні.

Після визначення терміну «домофон», давайте розглянемо різновиди домофонних ключів. Вони можуть бути класифіковані наступним чином: ключ-код, який потрібно вводити на інформаційній панелі; RFID-ключі; ключі Touch-Memory.

1.2 RFID технологія

RFID – це технологія, що використовується для домофонних ключів у формі невеликих брелоків або карток, які вже знайомі багатьом людям. Щоб активувати такий ключ, його потрібно піднести на певну відстань до домофона. В залежності від дальності спрацювання, ключі можуть бути класифіковані наступним чином: з зоною ідентифікації від 100 до 150 мм, що відповідає поширеному формату типу Proximity, або з дальністю визначення до 1 метра, відомі як ключі типу Vicinity.

Незважаючи на відмінності в дальності, робота всіх ідентифікаторів відбувається за простою схемою. Домофон, який використовує такі ключі, має блок, що випромінює слабке електромагнітне поле в області контактної площадки. Усередині RFID-карти або брелка розташована проста схема, яка містить індуктивний коливальний контур, мініатюрну передавальну антену та чіп, що генерує сигнал. При внесенні ключа в зону випромінювання відбувається активація внутрішньої електронної схеми та передача радіочастотного сигналу. Домофон розпізнає ідентифікатор та розблоковує двері, якщо ключ відповідає запису в його пам'яті. Зазвичай такі ключі працюють на частоті 125 Khz. і на частоті 13.56 МГц. такі як Mifare.

Такий ключ може вийти з ладу як через тріщини (що спричиняє пошкодження чіпа або передавальної сітки антени), так і внаслідок впливу інтенсивного електромагнітного випромінювання.

RFID-технології наступних типів широко використовуються: наприклад, старіший HID, популярний EM-Marin, а також картки Mifare, які використовуються для безконтактного використання на великій відстані.

Touch-Memory – це контактні таблетки, які вже знайомі більшості людей. Усередині такого ключа також розташований мікрочіп. Однак передача ідентифікатора відбувається через електричну одноканальну схему. Коли ключ доторкається до контактного майданчика, в домофоні відбувається замикання ланцюга зчитування даних. Унікальний код, запрограмований в таблетці, передається та перевіряється на відповідність одному з записів у пам'яті пристрою. Якщо відбувається відповідність, то двері розблоковуються.

Таблетку Touch-Memory можна пошкодити, вплинувши на неї сильним статичним зарядом, наприклад, приклавши її до наелектризованого одягу. Зробити це досить складно, оскільки імпульс повинен пройти через певні точки контактної площадки, але це є найпоширенішою причиною поломок. Таблетка з чіпом дуже міцна і механічне пошкодження її складно. Крім впливу статики, ключ можна пошкодити, використавши його в мікрохвильовій печі. Проте, Touch-Memory може витримати будь-які інші впливи, включаючи найпотужніші ніодімові магніти, без наслідків.

Таблетки Touch-Memory поділяються на різні класи, які визначаються виробниками. Кожен виробник створює внутрішню схему з певними характеристиками та методикою формування унікального коду. Широко використовуються такі типи Touch-Memory: ті, що мають маркування DS (Dallas) і використовуються в багатьох моделях, таких як Vizit, Eltis, C2000 та інші; ті,що мають маркування DC і використовуються в домофонах Cifral; а також серії К, які широко використовуються в системах контролю доступу Metacom та інших домофонах.

Тому, перед тим як дізнаватись, як програмувати особистий ключ для домофона під'їзної двері, важливо спочатку придбати Touch-Memory або RFID ключ сумісного формату.

1.3 Процес виготовлення дублікаторів

Для виготовлення домофонних ключів існує два варіанти. Перший варіант – звернутися до фірми, яка спеціалізується на домофонах у вашому місті. Вони зможуть зробити дублікат ключа за адресою, де розташований домофон. У них є бази даних, в яких зберігаються коди домофонних ключів.

Другий варіант – використання дублікатора, для якого необхідно мати робочий ключ. За допомогою дублікатора можна створити копію ключа.

Дублікатор домофонних ключів є невеликим портативним пристроєм, що працює на батарейках. Він дозволяє зчитати код з оригінального ключа і записати його на новий ключ-заготовку. Цей процес займає всього кілька секунд. Після виготовлення дубліката ключа, домофон буде реагувати на нього так само, як і на оригінальний ключ.

Важливо розуміти, що принцип роботи дублікатора базується на розпізнаванні та копіюванні коду з оригінального ключа. Це означає, що дублікатор може створити точну копію ключа з усіма його характеристиками. Цей метод дозволяє відтворити ідентичний ключ, який буде розпізнаватись домофоном і розблоковувати двері.

Застосування дублікатора домофонних ключів дозволяє вирішити проблему в разі втрати або поломки оригінального ключа. Завдяки ньому можна швидко і зручно створити дублікат ключа, що буде виконувати ті самі функції, що й оригінальний ключ, за умови що у вас є хоча б один робочий ключ.

Вибір між зверненням [7] до фахівців з домофонів або використанням дублікатора залежить від вашої ситуації та переваг. Звернення до фахівців може бути корисним, якщо ви остаточно загубили усі оригінальні ключі. З іншого боку, використання дублікатора може бути зручним і доступним варіантом, особливо якщо у вас вже є робочий ключ, з якого можна зчитати код.

Дублікатор може бути корисним як для створення ключів в невеликій кількості, також для задоволення великих замовлень у майстернях. Крім того, його можна використовувати спецслужбами для отримання доступу до під'їздів.

Існує багато готових дублікаторів [8], які відрізняються як у вартості, так і у функціональності. Проте, мій план полягає в створенні автономного пристрою з максимальними можливостями за мінімальну ціну, а також в описі процесу його монтажу.

РОЗДІЛ 2
ТЕОРЕТИЧНІ АСПЕКТИ ТА ВИБІР КОМПОНЕНТІВ

2.1 Складові дублікатора домофонних ключів

Стандартний контактний дублікатор [10], [11] домофонних ключів (рисунок 2.1) складається з наступних елементів:

- Корпусу, в якому знаходяться всі комплектуючі
- Кнопки перемикання режимів програмування.
- Контактної пластини.
- Світлодіода, який вказує на режим програмування.
- Живлення (можливе підключення через роз'єм Micro-USB або кронштейн).
- Дисплею, який виконує інформаційну функцію, так само як і світлодіод (необов'язково).
- Компонентів програмної плати та їхніх елементів.

Отже, давайте розглянемо компоненти, які будуть використовуватися під час збирання.

Рисунок 2.1 – Контактний дублікатор домофонних ключів [7]

2.2 Arduino

Arduino [12] – це апаратна платформа для розробки хобістських конструкцій, що складається з мікроконтролерної плати з вводом/виводом і розробницького середовища Processing/Wiring. Розробницьке середовище ґрунтується на спрощеній версії мов програмування C/C++. Arduino може бути використаний як для створення самостійних інтерактивних пристроїв, так і для з'єднання з програмним забезпеченням, що працює на комп'ютері, таким як Processing, Adobe Flash, Max/MSP, Pure Data, SuperCollider. (рисунок 2.2)

Рисунок 2.2 – Середовище Arduino [12]

Апаратна плата Arduino складається з мікроконтролера Atmel AVR та інтегрованих компонентів для програмування та з'єднання з іншими пристроями. Багато плат мають вбудований лінійний стабілізатор напруги, який забезпечує стабільне +5В або +3,3В живлення. Тактова частота плати може бути встановлена на 16 або 8 МГц, в залежності від типу кварцового резонатора. Завантажувач (bootloader) вже присутній у вбудованому мікроконтролері, тому зовнішній програматор не є необхідним.

На концептуальному рівні, всі платформи Arduino можуть бути програмовані через RS-232 (послідовне з'єднання), проте реалізація цього способу може відрізнятися в залежності від версії. У новіших платах використовується

програмування через USB, що можливо завдяки наявності мікросхеми конвертера USB-to-Serial, такої як FTDI FT232R. У версії платформи Arduino Uno в якості конвертера використовується контролер Atmega8 в SMD-корпусі. Це рішення дозволяє програмувати конвертер таким чином, щоб платформа могла бути визнана як миша, джойстик або інший пристрій, залежно від вибору розробника, з необхідними додатковими сигналами керування. У деяких варіантах, таких як Arduino Mini або неофіційна плата Boarduino, для програмування потрібно підключати окрему плату USB-to-Serial або використовувати спеціальний кабель.

Платформи Arduino дозволяють використовувати значну кількість виводів мікроконтролера як вхідні або вихідні контакти для підключення до зовнішніх схем. Наприклад, у платі Decimila доступно 14 цифрових входів/виходів, з яких 6 можуть генерувати сигнали ШІМ, а також 6 аналогових входів. Ці сигнали можна отримати на платі через контактні площадки або штирьові роз'єми. Крім того, існує багато різноманітних зовнішніх плат розширення, які називаються «shields» (щити), які можна підключати до платформи Arduino через штирьові роз'єми.

Arduino IDE є кросплатформовим додатком, розробленим на Java, який включає в себе кодовий редактор, компілятор і інструменти для завантаження прошивки на плату. Це середовище розробки засноване на мові програмування Processing і призначене для початківців, які мають обмежений досвід у розробці програмного забезпечення. Мова програмування Arduino подібна до мови Wiring і в основному базується на C++, з використанням деяких додаткових бібліотек.

Програми для Arduino обробляються спочатку препроцесором, а потім компілюються за допомогою AVR-GCC. Мови програмування, які використовуються для написання програм Arduino, включають C та C++. Середовище розробки Arduino поставляється з бібліотекою «Wiring», яка базується на проекті Wiring і спрощує багато стандартних операцій введення/виведення. Для створення програми, яка буде циклічно виконуватись, користувачам потрібно визначити дві функції:

setup(): Ця функція виконується лише один раз при запуску програми і використовується для встановлення початкових параметрів.

loop(): Ця функція виконується безперервно, доки плата не буде вимкнена, і виконує основну логіку програми.

2.3 Arduino Nano

Arduino Nano – це компактний пристрій, який має повну функціональність і базується на мікроконтролерах ATmega328 (Arduino Nano 3.0) або ATmega168 (Arduino Nano 2.x). Він спеціально розроблений для використання на макетних платах (рисунок 2.3). За своїми можливостями Arduino Nano подібний до Arduino Duemilanove, але відрізняється від нього розмірами та відсутністю роз'єму живлення. Також використовується інший тип USB-кабелю (Mini-B). Фірма Gravitech відповідає за розробку та випуск Arduino Nano.

Розміри і форм-фактор: Ардуіно Нано має розміри приблизно 4,5 см на 1,8 см і має прямокутну форму. Цей компактний розмір робить її дуже зручною для вбудовуваних систем, прототипів та проектів, де простір обмежений.

Arduino Nano може бути живлений за допомогою кабелю Mini-B USB або зовнішнього джерела живлення. Зовнішнє джерело живлення може мати нестабілізовану напругу в діапазоні від 6 до 20 В (з використанням виводу 30) або стабілізовану напругу 5 В (з використанням виводу 27). Автоматично вибирається джерело живлення з більшою напругою.

Важливою рисою Arduino Nano є те, що напруга на мікросхемі FTDI FT232RL подається лише у разі живлення пристрою через USB. Якщо Arduino Nano живиться від іншого зовнішнього джерела живлення (не через USB), то вихід 3.3В, що формується мікросхемою FTDI, буде неактивний. Це може призводити до мерехтіння світлодіодів RX і TX, особливо якщо на виводах 0 і 1 присутній високий рівень сигналу.

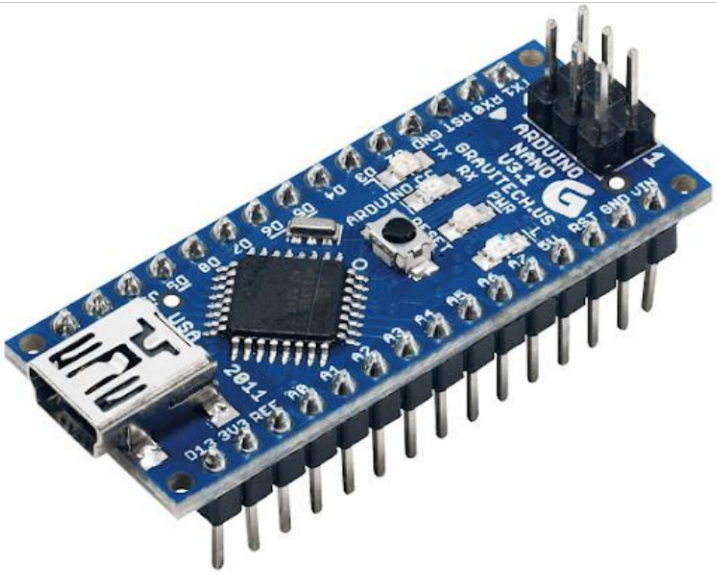

Рисунок 2.3 – Arduino Nano [12]

Мікроконтролер ATmega168 має обсяг програмної пам'яті 16 КБ, з яких 2 КБ використовуються завантажувачем. Окрім того, він має 1 КБ оперативної пам'яті SRAM і 512 байт EEPROM, для взаємодії з якою використовується бібліотека EEPROM.

У мікроконтролера ATmega328 обсяг програмної пам'яті становить 32 КБ, з яких 2 КБ відводяться під завантажувач. Також він має 2 КБ оперативної пам'яті SRAM і 1 КБ EEPROM.

Ці дані вказують на обсяг доступної пам'яті для зберігання програмного коду, змінних та даних на кожному з мікроконтролерів ATmega168 і ATmega328.

EEPROM (Electrically Erasable Programmable Read-Only Memory) є видом постійного запам'ятовувача, який може бути програмований та очищений за допомогою електричного сигналу. Це один з типів енергонезалежної пам'яті. EEPROM може бути циклічно очищено та заповнено інформацією багато тисяч разів. Цей тип пам'яті використовується в твердотільних накопичувачах. Одним з варіантів EEPROM є флеш-пам'ять.

Принцип дії EEPROM ґрунтується на заміні та авторизації електричного сигналу. Процес зміни заряду («запис» та «стирання») виконується шляхом подання великої напруги між затвором і витоком, створюючи високе електричне поле у тонкому діелектрику між каналом транзистора і кишенею, що викликає тунельний ефект. Для покращення ефективності тунелювання електронів у

кишеню під час запису використовується невелике прискорення електронів шляхом пропускання струму через канал полякового транзистора. Читання виконується за допомогою полякового транзистора, де кишеня виступає в ролі затвора. Зміна потенціалу на затворі змінює порогові характеристики транзистора, які реєструються ланцюгами читання.

Основною характеристикою класичного EEPROM є наявність додаткового транзистора, що допомагає управляти режимами запису та стирання. Деякі варіанти реалізації полягали у використанні одного тризатворного полякового транзистора (з одним плавним та двома звичайними затворами). Ця конструкція може бути використана в великому масиві подібних EEPROM завдяки використанню відповідних компонентів.

Підключення між ними здійснюється у формі двовимірної матриці, де кожна клітинка розташовується на перетині стовпчиків та рядків. Оскільки EEPROM має третій затвор, до кожної клітинки підходять три провідники (один для стовпчиків і два для рядків).

За допомогою функцій pinMode(), digitalWrite() і digitalRead() кожен з 14 цифрових виводів Arduino Nano може використовуватися як вхід або вихід. Робоча напруга виводів становить 5В. Максимальний струм, який може бути переданий або спожитий одним виводом, складає 40 мА. Всі виводи мають внутрішні резистори (за замовчуванням вимкнені) з номіналом 20-50 кОм. Крім основних функцій, деякі виводи Arduino можуть виконувати додаткові функції:

- Послідовний інтерфейс: виводи 0 (RX) і 1 (TX) використовуються для отримання (RX) і передачі (TX) даних через послідовний інтерфейс. Ці виводи пов'язані з відповідними виводами мікросхеми-перетворювача USB-UART від FTDI.
- Зовнішні переривання: виводи 2 і 3 можуть бути налаштовані як джерела переривань, які виникають при різних умовах, таких як низький рівень сигналу, фронт сигналу, спад сигналу або зміна сигналу.
- ШІМ: виводи 3, 5, 6, 9, 10 і 11 можуть генерувати аналогові сигнали у вигляді широтно-імпульсної модуляції (ШІМ) за допомогою функції analogWrite().

- Інтерфейс SPI: виводи 10 (SS), 11 (MOSI), 12 (MISO) і 13 (SCK) дозволяють здійснювати зв'язок за допомогою інтерфейсу SPI. У пристрої підтримується апаратна підтримка SPI, але на момент написання цієї відповіді мова Arduino цю функціональність ще не підтримує.

- Світлодіод: вивід 13 має вбудований світлодіод, який підключений до цифрового виводу 13. При подачі сигналу HIGH світлодіод буде ввімкнено, а при подачі сигналу LOW – вимкнено. У Arduino Ethernet є також 8 аналогових входів, кожен з яких може зчитувати аналогові напруги у вигляді 10-бітних значень (з 1024 різних рівнів). За замовчуванням вимірювання напруги здійснюється у межах діапазону від 0 до 5В. Проте верхню межу цього діапазону можна змінити, використовуючи вивід AREF (аналоговий опорний вивід) та функцію analogReference().

Крім зазначених, на платі Arduino Nano є ще кілька виводів зі своїми особливостями:

- Інтерфейс I2C: виводи 4 (SDA) і 5 (SCL) можуть бути використані для зв'язку за допомогою інтерфейсу I2C (TWI) за допомогою бібліотеки Wire. Цей інтерфейс дозволяє підключати різноманітні пристрої, які підтримують цей протокол зв'язку.

- AREF: це вивід, який може бути використаний як опорна напруга для аналогових входів. Використання функції analogReference() дозволяє змінити верхню межу діапазону вимірювання напруги на аналогових входах.

- Reset: цей вивід використовується для перезавантаження мікроконтролера. Формування низького рівня (LOW) на цьому виводі приведе до перезапуску мікроконтролера.

Таким чином, Arduino Nano (рисунок 2.4) надає широкі можливості для використання цифрових та аналогових виводів, а також підтримку різноманітних інтерфейсів, що дозволяє підключати його до різних пристроїв і розширювати його функціональні можливості.

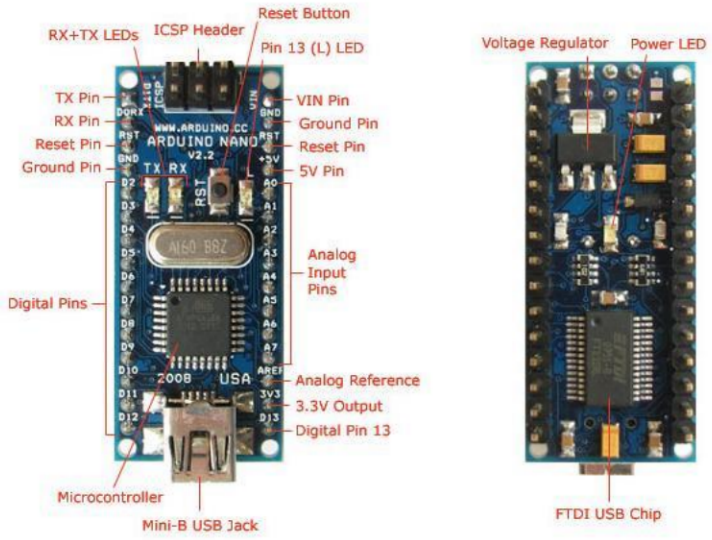

Рисунок 2.4 – Складові Arduino Nano [12]

З метою уникнення необхідності у кожен раз натискати кнопку скидання перед завантаженням програми, Arduino Nano був спроектований з можливістю програмного скидання з підключеного комп'ютера. Один з вихідних сигналів мікросхеми FT232RL, що відповідає за управління потоком даних (DTR), з'єднаний з входом RESET мікроконтролера ATmega168 або ATmega328 за допомогою конденсатора з ємністю 100 нФ. Коли лінія DTR переходить у нульовий рівень, вхід RESET також знижується на достатній проміжок часу для перезавантаження мікроконтролера. Ця функція була реалізована, щоб дозволити прошивку мікроконтролера за допомогою одного натискання кнопки в середовищі програмування Arduino. Така архітектура сприяє скороченню таймауту завантажувача, оскільки процес прошивки завжди синхронізований зі спадом сигналу на лінії DTR.

Однак, ця система може мати й інші наслідки. При підключенні Arduino Nano до комп'ютерів з операційними системами Mac OS X або Linux, його мікроконтролер буде скидатися при кожному з'єднанні з програмним забезпеченням. Після скидання на Arduino Nano запускається завантажувач протягом приблизно півсекунди. Незважаючи на те, що завантажувач запрограмований для ігнорування зовнішніх даних (тобто будь-яких даних, що не стосуються процесу прошивки нової програми), він може перехопити декілька перших байтів даних, що надсилаються платі відразу після встановлення

з'єднання. Тому, якщо програма, що працює на Arduino, передбачає отримання від комп'ютера будьяких налаштувань або інших даних при першому запуску, впевніться, що програмне забезпечення, з яким взаємодіє Arduino, надсилає ці дані через секунду після встановлення з'єднання. Таким чином, ви зможете уникнути перехоплення цих даних завантажувачем під час його активації після скидання мікроконтролера.

Враховуючи ці особливості, вам рекомендується у програмі Arduino забезпечити відправку налаштувань або інших необхідних даних через секунду після встановлення з'єднання з комп'ютером, щоб уникнути їх перехоплення завантажувачем.

Будьте уважні при роботі з Arduino Nano на комп'ютерах з операційними системами Mac OS X або Linux, і зверніть увагу на особливості програмування для досягнення бажаних результатів.

2.4 Зчитувач Touch-memory

Зчитувач Touch-memory, (контактна луза), призначений для отримання даних з контактного домофона та передачі їх до головного процесора. Цей зчитувач також використовується як записуючий елемент в дублікаторі домофонних ключів (рисунок 2.5).

Контактна луза має два контакти – «плюс» та «мінус» (розташовані на корпусі), які використовуються для зчитування інформації з пристрою.

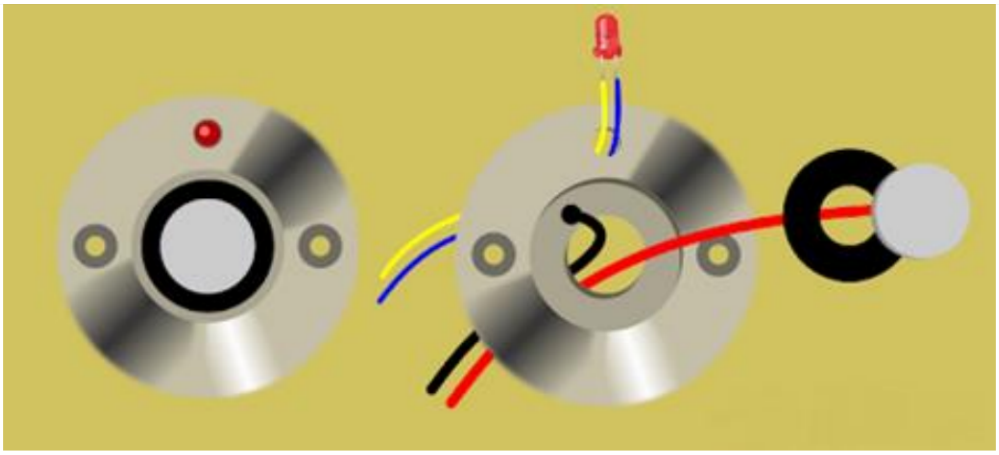

Рисунок 2.5 – Схема контактної лузи [11]

2.5 Резистор

Резистор [13] або опір (походить від латинського слова «resisto», що означає «опиратися») – це пасивний компонент електричного кола, який призначений для створення електричного опору (рисунок 2.6). Головною характеристикою резистора є його електричний опір. Закон Ома описує залежність між електричною напругою та електричним струмом, що протікає через резистор, у випадку лінійної характеристики. Він встановлює, що струм через резистор пропорційний напрузі, і співвідношення визначається значенням електричного опору.

Рисунок 2.6 – Види резисторів [11]

Основовне завдання резистора полягає у обмеженні сили струму, яка протікає через нього. Цей принцип роботи описується законом Ома: U = I x R, де U – напруга, I – сила струму, R – опір. Ом є одиницею вимірювання опору.

Резистор можна уявити як гнучкий водопровідний шланг (рисунок 2.7). Під тиском вода протікає через нього, але якщо встановити перешкоду на шляху води, наприклад, цеглу, діаметр трубки зменшиться, і вода буде витікати меншим потоком. Аналогічно резистор обмежує силу струму: його величина зменшується під час протікання через резистор.

Коли струм проходить через резистор, його значення знижується. Це означає, що частина електричної енергії, яка протікає через резистор, перетворюється на теплову енергію.

Наприклад, якщо підключити світлодіод до батарейки без резистора, він швидко перегріється і вийде з ладу, оскільки сила струму, яка протікає через нього, буде надто великою. Тому резистор є важливим компонентом в електроніці для захисту інших елементів від надмірного струму.

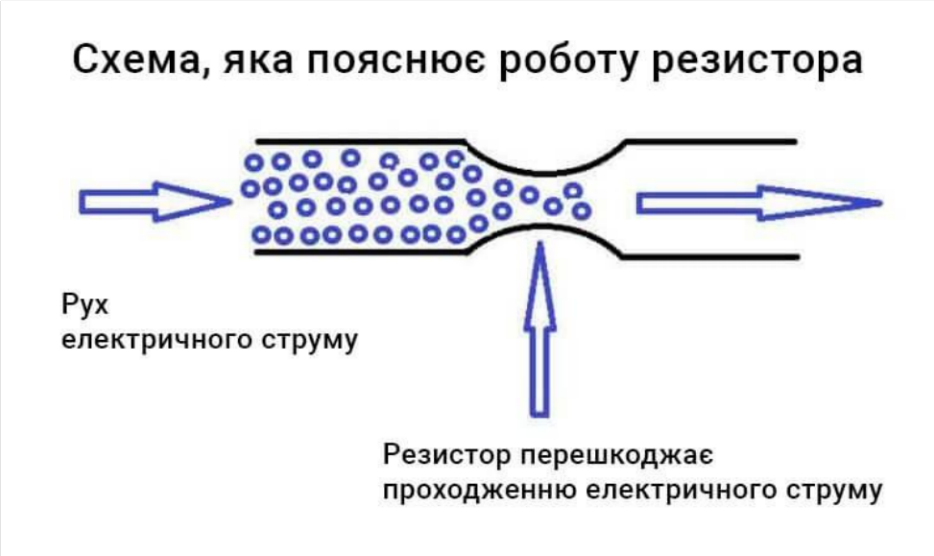

Рисунок 2.7 – Принцип роботи резистора [13]

Резистори відносяться до електронних елементів, що застосовуються в схемах електротехніки та електроніки для обмеження потоку струму та розподілу напруги. Резистори є найпоширенішими пасивними компонентами електронної апаратури, що використовуються як навантаження, споживачі та роздільники в колах живлення, як елементи фільтрів, шунти, в колах формування імпульсів та інших пристроях.

Резистори характеризуються номінальним значенням електричного опору (від декількох омів до кількох гігоомів), прийнятним відхиленням від нього (0,001...20%), максимальною потужністю розсіювання (від дрібних ватт до сотень ватт), межовою електричною напругою та температурним коефіцієнтом електричного опору.

Залежно від призначення, резистори можна розділити на дві категорії: резистори загального призначення та резистори спеціального призначення, до яких входять високоомні резистори, високовольтні резистори, високочастотні резистори та прецизійні резистори.

За матеріалом, який використовується для створення опору, резистори класифікуються на наступні типи:

- Дротяні резистори: це найдавніший тип резисторів, які складаються з відрізка дроту з високим опором, що намотаний на неметалевий каркас. Вони можуть мати паразитну індуктивність.
- Плівкові металеві резистори: цей тип резисторів складається з тонкої металевої плівки з високим опором, що нанесена на керамічне осердя. На кінцях осердя розміщені металеві ковпачки з дротяними виведеннями. Це найпоширеніший тип резисторів.
- Металофольгові резистори: вони використовують тонку металеву стрічку як резистивний матеріал.
- Вугільні резистори: ці резистори можуть бути плівковими або об'ємними. Вони використовують високий опір графіту.
- Напівпровідникові резистори: вони використовують слабколегований напівпровідник як матеріал для створення опору. Ці резистори можуть мати як лінійну, так і нелінійну вольт-амперну характеристику. Вони переважно використовуються в складі інтегральних мікросхем, де застосування інших типів резисторів є складнішим.

Залежно від зміни опору, резистори можуть бути розділені на такі категорії: резистори з постійним опором, регульовані резистори зі змінним опором (потенціометри) та підлаштовні резистори зі змінним опором. Для мого проекту потрібні типові резистори з дротяними аксіальними виводами для навісного монтажу, з номінальним значенням 2.2 кОм.

2.6 Енкодер

Спочатку я хотів використати звичайний перемикач у своєму проекті, але енкодер [14] являється кращим рішенням. У контексті Arduino, датчики обертання (Енкодери) (рисунок 2.8) зазвичай використовуються разом з руховими системами, такими як сервомотори або крокові двигуни, для контролю їхнього положення або обертання. Датчик обертання може вимірювати кількість обертів або зміну положення рухомого об'єкта і передавати цю інформацію до мікроконтролера Arduino.

Рисунок 2.8 – Енкодер для Arduino [14]

Існує кілька типів датчиків обертання, включаючи оптичні та магнітні. Оптичний датчик обертання має спеціальні сенсори, які реагують на переривання світла і генерують електричні сигнали, що вказують на зміну положення об'єкта. Магнітний датчик обертання використовує магнітні поля для вимірювання руху.

При роботі з датчиками обертання на Arduino, необхідно підключити їх до вхідних/вихідних портів мікроконтролера і використовувати програмне забезпечення (скетч) для зчитування сигналів, що генеруються датчиком. За допомогою програми можна інтерпретувати ці сигнали та виконувати певні дії на основі зміни положення або обертання об'єкта.

Наприклад, можна використовувати датчик обертання для керування положенням робочої головки на 3D-принтері, або перемикачем режимів на дублікаторі домофонних ключів.

2.7 Батарея крона

Для забезпечення електроенергією всієї системи нам потрібне надійне джерело живлення з напругою 5 В, наприклад, таке як батарейка типу «Крона» [15] (рисунок 2.9).

Батарейка типу «Крона» належить до категорії батарейок стандартного розміру. Назва походить від назви вуглецево-марганцевих батарейок цього самого розміру, що вироблялися в колишньому СРСР.

Батарейка типу «Крона» має номінальну ємність 0,5 ампер-години. При робочій напрузі 8,4 В, свіжезаряджений акумулятор може декоротий час надавати напругу 11,5 В або ще вищу, що пояснюється особливостями Ni-MH акумуляторних елементів, що складають батарею.

Протилежно до хімічних джерел живлення інших типів, звичайні одноразові батареї «Крона» мали обмежену можливість дозарядки, навіть якщо це не рекомендувалося виробником. Тому в багатьох журналах і книгах, таких як «Енциклопедія юного техніка» 1977 року, публікувалися схеми зарядних пристроїв, що надавали особливо формований струм. Ці схеми зарядних пристроїв набули певної популярності через дефіцит батарей.

Рисунок 2.9 – Батарея крона [15]

2.8 Інструменти для монтажу та пайки

Паяльник – простий ручний інструмент, який використовується для нагрівання об'єктів шляхом передачі тепла від нагрітого металевого елемента. Використовується для лудіння поверхонь та спаювання деталей шляхом точкового паяння. Складається з робочого елемента («жала»), який нагрівається, і теплоізольованої ручки. Типові паяльники призначені для використання з м'якими паяльними сплавами.

Паяльна станція [16] – багатофункціональний стаціонарний паяльний інструмент, який застосовується в галузі електроніки та електротехніки (рисунок 2.10). Вона дозволяє здійснювати паяння чутливих електронних компонентів з дотриманням усіх технічних регламентів, що стосуються температури, тривалості паяння, рівномірності та швидкості нагрівання, розміру зони нагрівання і т. д. Паяльна станція забезпечує точний контроль температури і зручність в роботі.

Рисунок 2.10 – Паяльна станція [16]

Припій – це метал, сплав або суміш оксидів, який використовується для з'єднання металевих, мінерало-керамічних та інших деталей, а також для лудіння посуду та інших застосувань (рисунок 2.11). Процес з'єднання деталей за допомогою припою називається паянням. Під час паяння рідкий припій

заповнює проміжок між деталями, і після затвердіння він утворює міцне з'єднання. Температура плавлення припою зазвичай нижча, ніж у самих деталей, що дозволяє йому легко розтопитися і з'єднати деталі.

Рисунок 2.11– Припій [16]

Флюс – це речовина, яка додається до розплавленого металу для видалення окисів та сторонніх шлаків або для запобігання окисненню поверхні металу під час паяння (рисунок 2.12). Вибір флюсу залежить від температури плавлення металу або температури паяння. Деякі з найпоширеніших типів флюсів включають вапняк, силікати, буре залізо, борну кислоту і каніфоль. Флюси допомагають забезпечити чисту поверхню металу для стабільного з'єднання та зменшують утворення окисних плівок під час паяння.

Рисунок 2.12 – Паяльний флюс MECHANIC 225 [16]

Корпус для мого виробу я взяв пластиковий за низькою ціною. Для з'єднання компонентів виробу я використовую спеціальні провідники зручної

конструкції, що мають сумісний роз'єм для підключення до Arduino Nano. (рисунок 2.13). Також відрізавши контакти ми отримуємо відмінні проводи для пайки і електричного з’єднання.

Рисунок 2.13 – Проводи для Arduino

Термоусадкова гофрована трубка [17] – пластиковий гнучкий елемент, що призначений для електроізоляції, механічного захисту з'єднань провідників, паяння та інших з'єднань (рисунок 2.14). Вона також може бути використана для відновлення пошкодженої ізоляції кабелів, захисту їх від механічних ушкоджень, забезпечення герметичності ізоляції під час обробки кабелю та інших застосувань.

Рисунок 2.14 – Термоусадка [17]

2.9 RFID компоненти для зчитування

Модуль Arduino RC522 – це RFID-считувач, що базується на мікросхемі MFRC522 (рисунок 2.15). Він дозволяє зчитувати та записувати дані на RFID-карти та RFID-мітки. Цей модуль надає можливість використання технології безконтактної ідентифікації, що дозволяє втілювати різноманітні застосування, такі як контроль доступу, системи безпеки, інвентаризація та багато іншого.

Основні характеристики модуля Arduino RC522:

- Інтерфейс: SPI (Serial Peripheral Interface).
- Робоча частота: 13.56 МГц.
- Підтримувані стандарти: ISO/IEC 14443 A/MIFARE.
- Считування та запис даних на RFID-карти та мітки.
- Підтримка різних типів міток, таких як MIFARE Classic 1K, MIFARE Classic 4K, і т.п.
- Відстань зчитування: до 3-5 см.

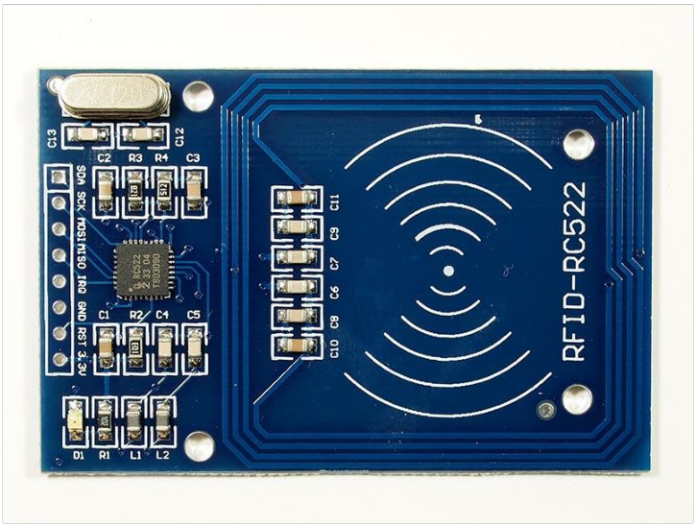

Рисунок 2.15 – Зчитувач Arduino RC522 [12]

Модуль Arduino RC522 має зручний набір функцій та бібліотек, що спрощує роботу з ним. Щоб використовувати цей модуль, потрібно підключити його до вашої Arduino плати за допомогою інтерфейсу SPI та використовувати відповідну бібліотеку для считування та запису даних на RFID-карти та мітки. Єдиним його мінусом є те що він може лише зчитувати і обробляти інформацію,

але записувати RFIDмітки автономно від ПК він не може. Що нам не дуже підходить.

Котушка для зчитування RFID міток (рисунок 2.16) на частоті 125 кГц використовується для безконтактної ідентифікації та взаємодії з RFID-мітками. Вона генерує електромагнітне поле потрібної частоти та отримує відповідні сигнали від міток.

Рисунок 2.16 – Котушка з модулем [11]

Основна роль котушки полягає у створенні високочастотного сигналу на визначеній частоті та прийманні відповідних сигналів від RFID-міток. Вона складається з провідника, намотаного на спеціальну раму або бобіну. Конфігурація та кількість витків котушки можуть змінюватися відповідно до потреб системи та установки.

Підключивши котушку до RFID-считувача, вона створює змінне магнітне поле, яке проникає в навколишнє середовище. RFID-мітки, розташовані в цьому полі, отримують енергію від поля та починають передавати свою ідентифікаційну інформацію назад до считувача. Котушка також виступає як антена, яка приймає сигнал від міток та передає його до считувача для подальшої обробки.

Частота 125 кГц використовується у багатьох типах RFID-систем, таких як системи контролю доступу, системи безпеки тощо. Вона має добру проникність через матеріали, які не обробляються, наприклад, пластик та дерево, але меншу проникність через метал та воду.

РОЗДІЛ 3
ПРАКТИЧНА РЕАЛІЗАЦІЯ ПРОЄКТУ

3.1 Схема проекту

Для початку роботи над проектом варто розробити схему (план), яка буде служити основою. Тому першим кроком буде підготовка схеми (рисунок3.1), яку можна зобразити у програмі Paint 3D, яка передбачена для встановлення на ОС Windows 10. Програма є досить простою у використанні: необхідно знайти необхідні компоненти в Інтернеті та об'єднати їх у єдину схему.

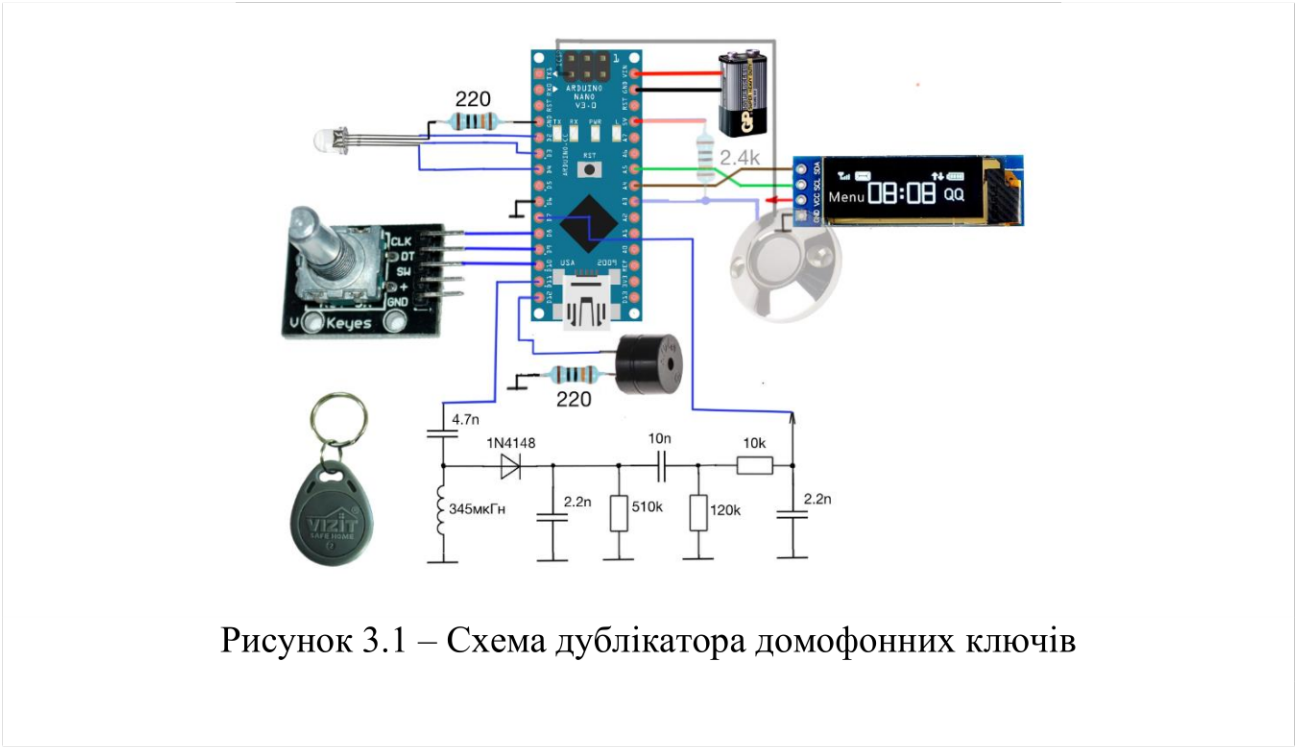

Рисунок 3.1 – Схема дублікатора домофонних ключів

Після підготовки схеми наступним кроком є розробка скетчу (програмного коду), який буде виконуватися на платформі Arduino. Для цього можна скористатися Arduino IDE (Integrated Development Environment) – інтегроване середовище розробки, призначене спеціально для програмування Arduino.

3.2 Пайка компонентів Arduino Nano

Розкладемо спочатку всі вище підібрані компоненти (рисунок 3.2). Не включаючи: паяльну станцію , припій , термоусадку і котушку.

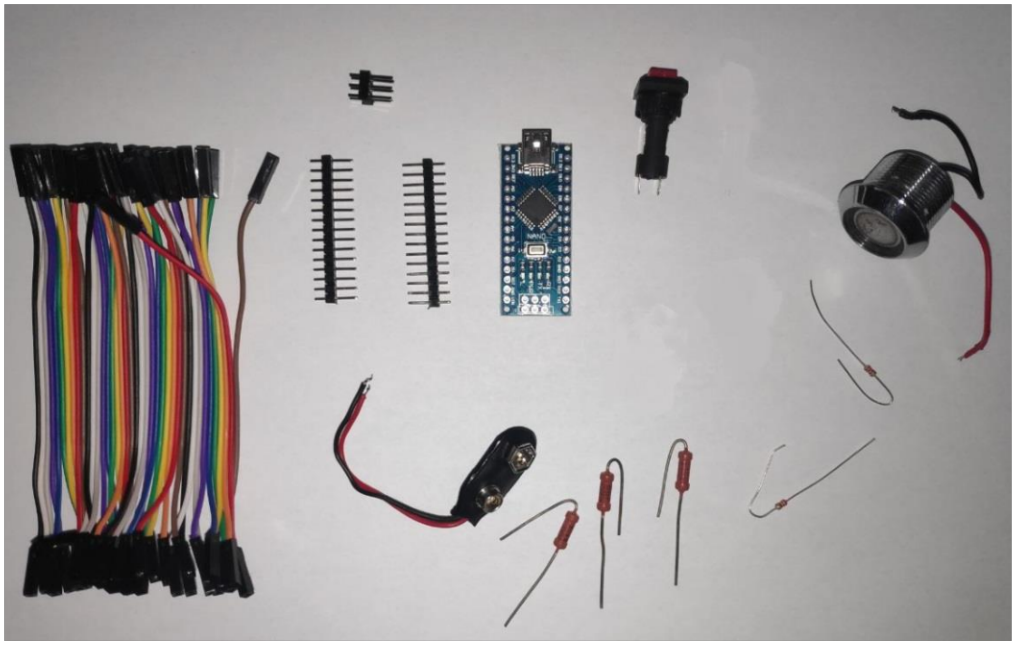

Рисунок 3.2 – Компоненти для дублікатору домофонних ключів

А також котушку для зчитування RFIDміток (рисунок 3.3). В інтернеті вдалося знайти готову котушку, але вона була на 420 мкГн., а мені потрібно на 345 мкГн., відмотавши, декілька вітків від котушки я зміг добитися робочої величини. В перевірці мені допоміг LC-метр.

Рисунок 3.3 – Котушка для зчитування RFID-міток

Спочатку потрібно підготувати Arduino, для початку вмикаємо паяльник для розігріву та припій (рисунок 3.4).

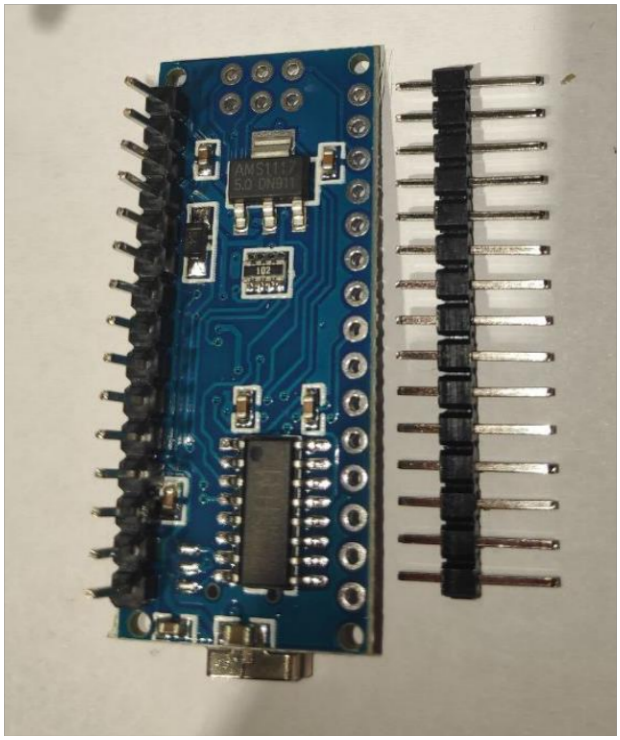

Рисунок 3.4 – Вставляю контакти в Arduino Nano

Далі потрібно припаяти контакти до Arduino Nano, для цього я використовую паяльний флюс в сумісності з оловом (рисунок 3.5), і за допомогою паяльника (паяльної станції).

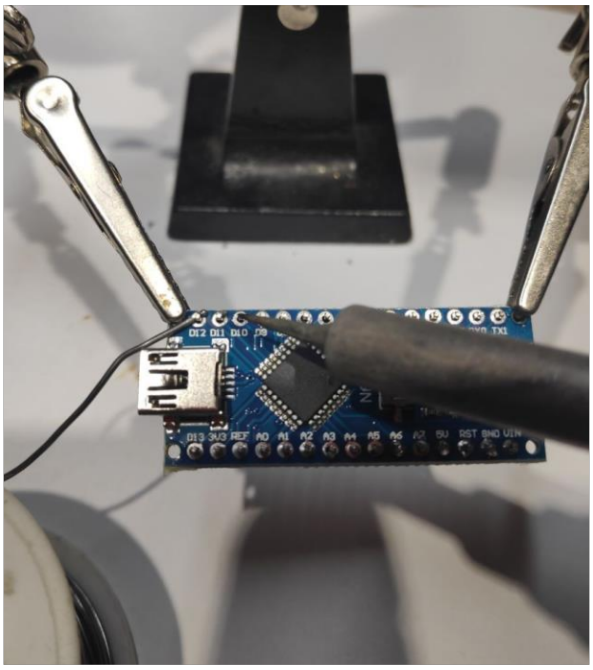

Рисунок 3.5 – Пайка контактів Arduino Nano

3.3 Пайка компонентів програматора

В першу чергу я припаяю проводи до контактної лузи. Відрізаю один кінець провода та зачишчаю. Відмірюю потрібну довжину термоусадки та відрізаю. Одягаю термоусадку на провід від контактної лузи так щоб вона не заважала пайці (рисунок 3.6).

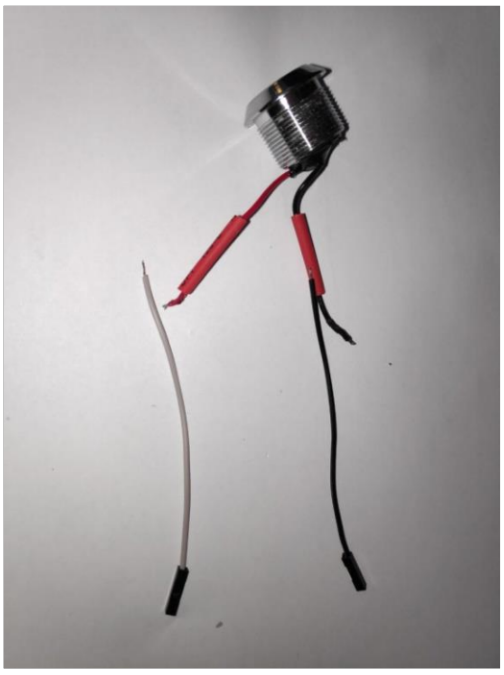

Рисунок 3.6 – Підготовка до пайки

З’єднуємо проводи та фіксуємо термоусадку за допомогою гарячого повітря, з паяльної станції, або за допомогою звичайної запальнички (рис. 3.7). Якщо не має можливості купити контактну лузу, тоді можна використати дві звичайні мідні пластини, які будуть виконувати функції «плюса» та «мінуса». Такий віріант буде набагато дешевше, але не практичніше, тому я вибрав саме контактну лузу.

Таким самим чином підпаюю контакти для крони, енкодера, а також дисплея,та ховаю в термоусадку (рисунок 3.8).

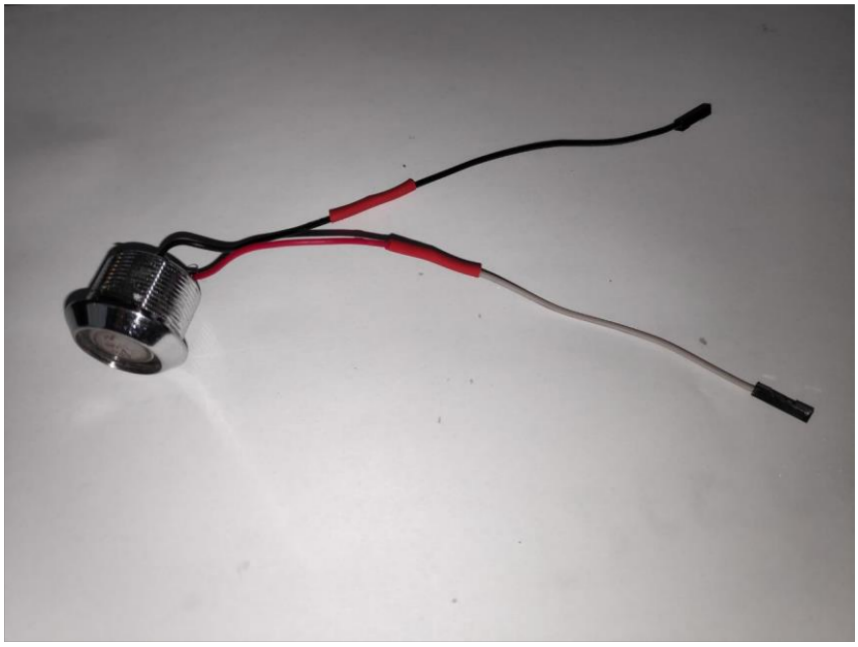

Рисунок 3.7 – Готова контактна луза

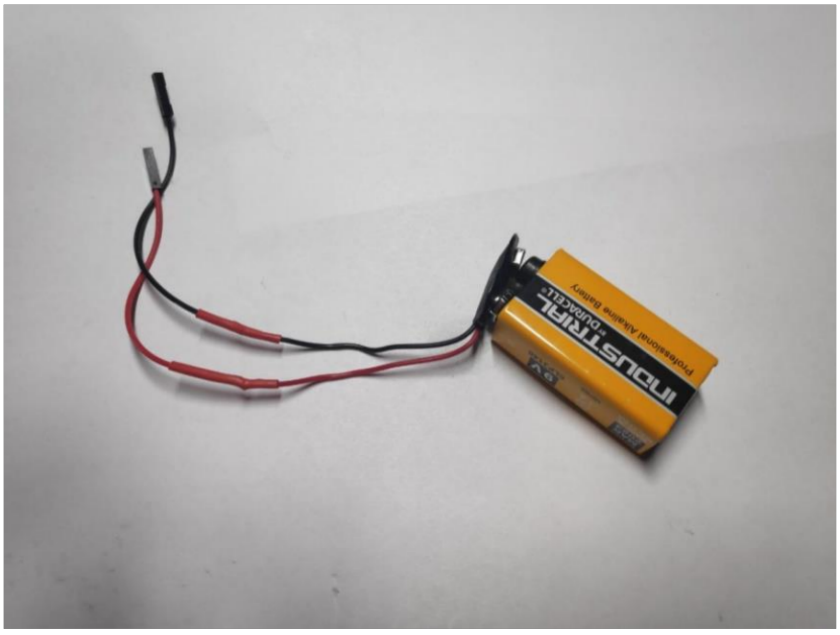

Рисунок 3.8 – Готовий конектор для крони

Щоб підпаяти проводи до кнопки, я спочатку їх наживив в отвори на кнопці, обмотав навколо, і припаяв припоєм.

Також по схемі є декілька резисторів, їх я теж припаюю і ізолюю термоусадкою. Для роботи катушки потрібно виготовити модуль (рисунок 3.9) який буде ізолювати усі шуми і приймати інформацію.

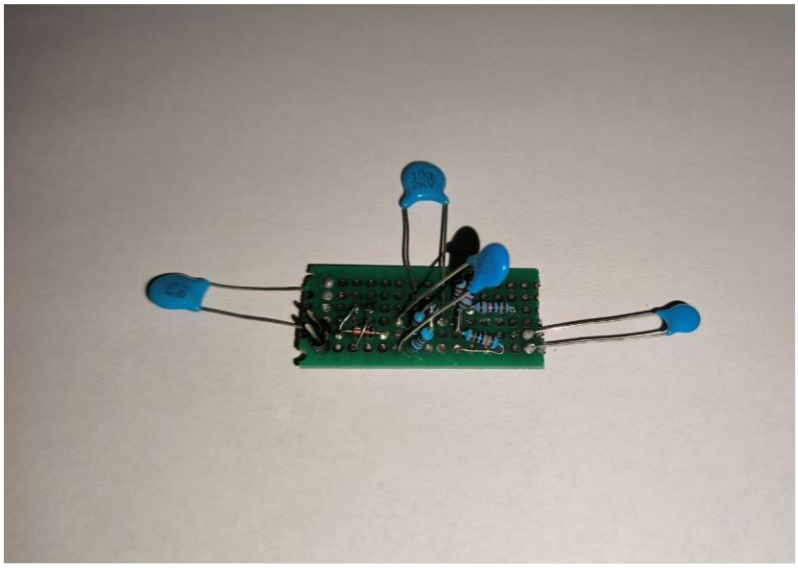

Рисунок 3.9 – Модуль для зчитування RFID-міток

3.4 Монтаж компонентів

Підпаявши усі компоненти (рисунок 3.10), мені залишилося підключити компоненти до ArduinoNano, та помістити усе в корпус.

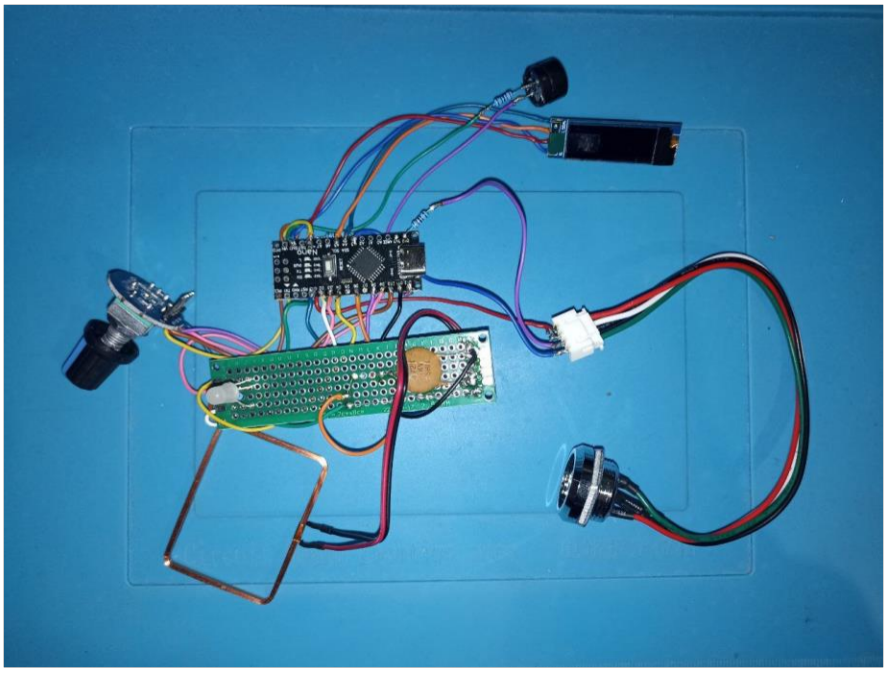

Рисунок 3.10 – Зібрані усі компоненти

Спочатку підготовлюю корпус для монтажу, а саме роблю отвори для контактної лузи, екрану та енкодера. Для крони місце уже є в корпусі. Вставляю компоненти на свої місця та фіксую термоклеєм.

ArduinoNanoфіксую термоклеєм, так щоб був доступ до порту mini-usb. Крону вставляю в спеціально відведене місце, підключаю усі компоненти згідно схеми та роблю мінімальний кабель-менеджмент. Закриваю корпус та фіксую на само різи (рисунок 3.11).

Рисунок 3.11 – Зібраний дублікатор домофонних ключів

3.5 Загрузка скетча

Для загрузки скетча потрібно встановити додаток ArduinoIDE, який можна завантажити з офіційного сайта в ZIP-форматі. Встановлюю драйвера на ArduinoNano, відкриваю програму та прописую скетч. Тепер в меню «Інструменти» вибираю мою плату, та порт до якого вона під'єднана, а також встановлюю бібліотеку «OneWire», що дозволяє здійснювати передачу даних. Натискаю вивантажити, і починається компілювання скетчу, через деякий час мені пише що все пройшло успішно, і дублікатор готовий до роботи. (рис. 3.12)

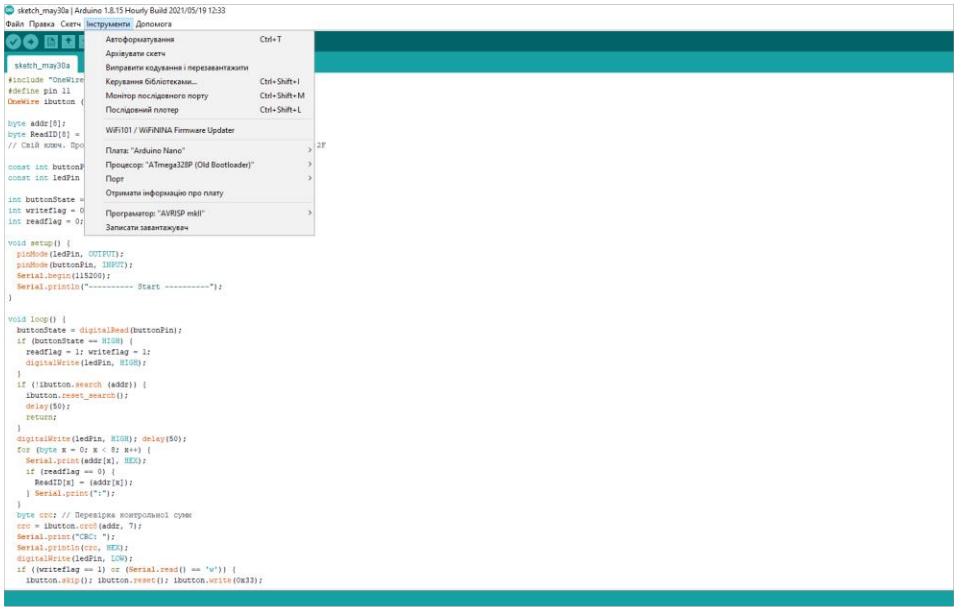

Рисунок 3.12 – Скетч та налаштування в Arduino IDE

3.6 Перевірка роботи дублікатора

У мене є синя заготовка – з кодом від домофона (рисунок 3.13).

Рисунок 3.13 – Зчитування контактного ключа

І у мене є зелена заготовка на яку я записав код з робочого ключа, перемкнувши, програматор у режим запису за допомогою енкодера (рис. 3.14).

Рисунок 3.14 – Дублікат контактного ключа

В моєму програматорі є три режими, а саме: Режим читання який вмикається замовчанням (горить зелений світлодіод), режим запису який вмикається якщо натиснути на енкодер (горить червоний світлодіод). І режим роздачі сигналу (горить синій світлодіод), який емулює код ключа зчитаного раніше, або збереженого в пам’яті програматора. Збереження коду ключа виконується при натисненні енкодера на 2 секунди. Можна зберегти до 20-ти кодів, вибрати потрібний код можна за допомогою енкодера. Коли пробуємо зберегти новий код, то самий старий з них видаляється, якщо у пам’яті програматора уже 20 кодів. Розглянемо дублювання безконтактного ключа (рисунок 3.15).

Рисунок 3.15 – Зчитування коду RFID-мітки

Для зчитування коду, підносимо заготовку до лузи на 2-3 секунди, так щоб заготовка доторкалася, і до центру лузи і до краю (світлодіод швидко моргає). Тепер перемикаємо дублікатор в режим запису і прикладаємо заготовку яку хочемо запрограмувати, потім чекаємо 4-5 секунд. Дублікат виготовлено успішно. Або якщо хочемо скопіювати RFID-мітку, прикладаємо її в зону для зчитування (рисунок 3.16).

При ввімкненні дублікатора в пам’яті машини висвітлюється код який прописаний в скетчі. Якщо перемкнути дублікатор в режим запису, і записати код в болванку, то ми отримаємо саморобний ключ до домофона. Код можна загрузити будь-який, який захочеться. Потім цей код програмується в EEPROM домофона, і ми отримуємо робочий домофонний ключ з власним кодом.

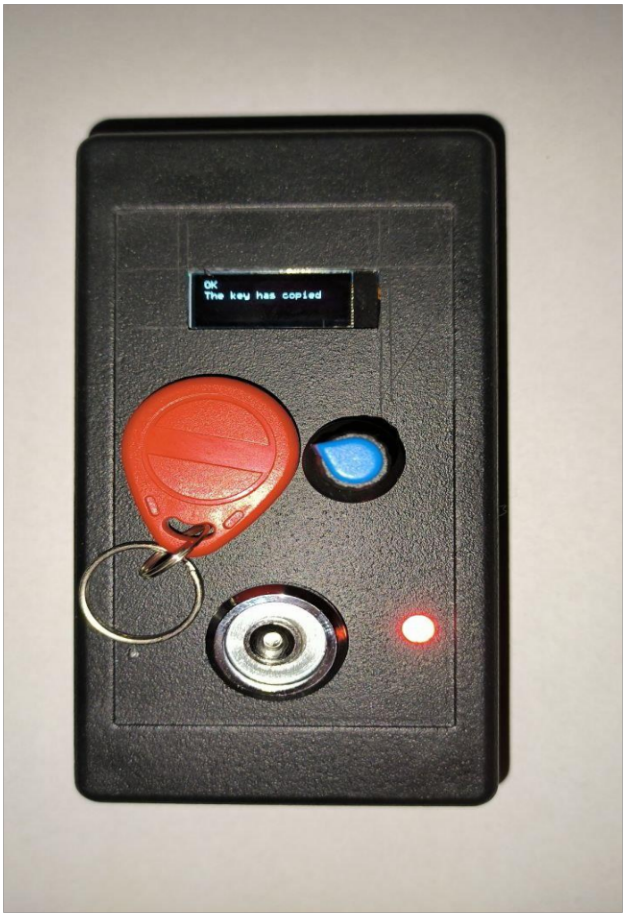

Рисунок 3.16 – Дублікат виготовлено успішно

Функція запису власного кода буде корисна для спецслужб. Перед виїздом на виклик, де встановлений домофон, один із співробітників спецслужби відкриває спеціальну базу даних, та вписує адресу домофона. База покаже код який прописаний на даному домофоні. Цей код копіюється та вставляється в скетч дублікатора домофонних ключів; скетч перезаписується, і тепер ми виготовляємо ключ, який підходить до вказаного домофона. Тому можна розвіяти міф про універсальні домофонні ключі. Ключ може підходити до різних домофонів тільки тоді, коли в них прописаний одинаковий код.

3.7 Розрахунок витрат на створення проектного пристрою

В даному розділі розраховано економічну доцільність впровадження розробки кваліфікаційної роботи, вході якого будуть проаналізовані витрати на розробку та впровадження пристрою, а саме рамки ефекту сповільнення часу базі платформи Arduino.

Сумарні витрати на розробку та організацію впровадження даного пристрою розраховуються за формулою:

$$W = W_1 + W_2 + W_3 + W_4, \quad (3.1)$$

де W_1 – матеріальні витрати;

W_2 – витрати на оплату праці;

W_3 – витрати на соціальне страхування;

W_4 – амортизація основних засобів.

Матеріальні витрати (W_1) – це витрати на сировину, матеріали, купівельні напівфабрикати та комплектуючі вироби, придбані у сторонніх організацій енергію, запасні частини, використані в операційній діяльності. Складовими матеріальних витрат у нашій роботі є: комплектуючі, що входять до складу пристрою, розхідні матеріали, а також вартість спожитої електроенергії для роботи, які продемонстровано в таблиці 3.1.

Таблиця 3.1 – Комплектуючі, що входять в склад пристрою

Найменування обладнання	Ціни за одиницю товару, грн	Кількість, шт.	Сума витрат, грн
Плата Arduino Nano V3.0	80	1	80
Коннектор крони	10	1	10
Контактна луза	100	1	100
Кнопка перемикання	10	1	10
Резистор 2.2кОМ	10	2	20
Крона	60	1	60
З’єднувальні провідники	1	20	20
Пластиковий корпус	40	1	40
Всього, $W_к$			340

Вартість розхідних матеріалів ($W_р$ (паяльник+каніфоль+клей+припой)) складає 350 грн.

При розробці проекту, мали місце витрати пов’язані з затратою ресурсів на нагрівання паяльника, роботу персонального комп’ютера, а саме затрати на електроенергію. Обчислимо ці затрати за наступною формулою:

$$W_е = N * S_{м.г} * t_{вит}, \quad (3.2)$$

де N – кількість робочих місць (N=1);

$S_{м.г}$ – вартість однієї машинно-годинної роботи робочого місця, грн/год.;

$t_{вит}$ – машинний час, витрачений на розробку даного проекту, год.

Споживання електроенергії при виконанні проекту становить орієнтовно 0,8 кВт*год (відповідно для промислових споживачів, другий клас напруги, ціна за 1 кВт становить 2,75 грн.

$$S_{м.г} = 0,8* 2,75= 2,2 \text{ грн/год}$$

Тривалість розробки проекту 4 години. Отже, вартість спожитої електроенергії при виконанні проекту становить:

$$W_e = 1 * 2,2 * 4 = 8,8 \text{ грн.}$$

Отже, сумарні матеріальні витрати дорівнюють:

$$W_1 = W_к + W_p + W_e = 340+350+8,8=698,8 \text{ грн.}$$

Розрахунок витрат на оплату заробітної плати (W_2*).* Заробітна плата – це винагорода, розрахована, як правило, у грошовому виразі, яку за трудовим договором роботодавець виплачує працівникові за виконану ним роботу. Розмір заробітної плати залежить від умов виконаної роботи, спеціальності розробника, що приймають участь в процесі проектування, тарифний розряд та трудомісткість робіт. Затрати на заробітну плату робітників здійснюються відповідно до спеціальності робітників, що приймають участь у процесі розробки, та на основі тарифних розрядів, а також трудомісткості робіт.

Розробниками даного проекту є працівники – електротехніки, які займаються паяльними роботами, також в їхні обов’язки входить виготовлення пошукової голівки та виготовлення корпусу. Відповідно до даних, які отримані у роботодавців, місячний оклад електротехніка, який займається пайкою, в

середньому складає 7000 грн. Місячний оклад програміста мікроконтролерів в середньому складає 10000 грн.

Виходячи з цих даних, можна вирахувати денну заробітну плату робітників ($ЗП_д$), яка вираховується за формулою:

$$ЗП_д = \frac{O_M \cdot (1+k)}{Ф_р}, \quad (3.3)$$

де, $О_м$ – місячний ставка;

k – коефіцієнт, що визначає розмір додаткової заробітної плати (приймаємо 1,25 для даної категорії працівників).

$Ф_р$ – кількість робочих днів у місяці. В середньому у місяці 22 робочих дні.

Тепер підставимо дані у формулу, і вирахуємо денну заробітну плату:

- для електротехніка:

k=1,25, $О_м$= 7000 грн, $Ф_р$ = 22

$$ЗП_д = \frac{7000 \cdot 1{,}25}{22} = 397{,}7 \text{ грн.}$$

- для працівника, який займається програмуванням Arduino:

k=1,25, $О_м$= 10000 грн, $Ф_р$ = 22

$$ЗП_д = \frac{10000 \cdot 1{,}25}{22} = 568{,}18 \text{ грн.}$$

Тривалість роботи над даним проектом електротехніка складає 1 год. протягом одного робочого дня, а працівника, який займається програмуванням – 4 години протягом одного робочого дня, то потрібно розрахувати вартість однієї години при тривалості робочого дня 8 годин:

$$ЗП_г = \frac{ЗПд}{Т}, \quad (3.4)$$

де $ЗП_д$ – денна заробітна плата робітника, грн., Т – тривалість робочого дня, який становить 8 год. Отже, годинна ставка дорівнює:

- для електротехніка:

$$ЗП_г = \frac{397,7}{8} = 49,71 \text{ грн/год}$$

- для працівника, який займається програмуванням Arduino:

$$ЗП_г = \frac{568,18}{8} = 71,02 \text{ грн/год}$$

Витрати на оплату праці (таблиця 3.2), враховуючи погодинну оплату працівизначаються за формулою:

$$ЗП = n \cdot t \cdot ЗПг, \quad (3.5)$$

де, *n* – чисельність робітників, чол.;

t – час, який затрачений на розробку проекту, годин;

ЗПг – заробітна плата за одну годину, грн.

Отже, витрати на оплату праці:

- для електротехніка:

$$ЗП_е = 1*1*49,71 = 49,71 \text{ грн.}$$

- для працівника, який займається програмуванням Arduino:

$$ЗП_п = 1*4*71,02 = 284,08 \text{ грн.}$$

Таблиця 3.2 – Розрахунок витрат на оплату праці

Спеціаліст	К-сть. чоловік	Час роботи, год.	Денна ЗП, грн/год.	Витрати на оплату, грн
Електротехнік	1	1	49,71	49,71
Програміст	1	4	71,02	284,08
Всього (W_2):				*333,79*

Наступним етапом в розрахунку сумарних витрат є *визначення витрат на соціальне страхування (*W_3). Ставка єдиного соціального внеску складає 22% від величини заробітної плати. Таким чином, ЄСВ:

$$W_3 = W_2 * 22\% \quad (3.6)$$

Отже, W_3= 333,79*22% = 73,43 грн.

Розрахунок амортизації основних засобів (W_4) здійснюється відповідно до діючих нормативно-правових документів щодо амортизації основних засобів у таблиці 3.3.

Таблиця 3.3 – Вартість амортизації апаратних засобів

№ п/п	Об'єкт Амортизації	Первісна вартість, грн.	Термін корисного використання обладнання, років	Сума амортизаційного відрахування,грн
1.	Комп'ютер	7200	5	120
Всього (W_4)		*7200*	-	*120*

Отже, де W_1=698,80 грн.

W_2 =333,79 грн.

W_3 =73,43 грн.

W_4 =120 грн.

Сумарні витрати, пов'язані із організацією проекту дорівнюватимуть:

$$W = 698,80+333,79+73,43+120=1226,02 \text{ грн.}$$

На розробку даного проекту і його впровадження було витрачено 1226, 02 грн. Для того, щоб порівняти дану розробку із цінами аналогічних пристроїв, слід розрахувати її ціну. Приймемо показник рентабельності на рівні 15%. Отже, ціна розробки складатиме:

$$Ц= W*P, \tag{3.7}$$

де P=15%

$$Ц=1226,02*1,15=1409,92 \text{ грн.}$$

Якщо порівняти вказану ціну з цінами аналогічних пристроїв (від 1700 грн) можна побачити, що розробка даного проекту економічно вигідною. З урахуванням даної ціни, економія від впровадження даного пристрою складатиме:

$$E = 1700 – 1409,92 = 290,08 \text{ грн.} \tag{3.8}$$

Для оцінки економічної ефективності визначимо термін окупності та коефіцієнт економічної ефективності. Термін окупності $T_{ок}$ розраховується за формулою:

$$T_{ок}=\frac{W}{E}=\frac{1226,02}{290,08}=4,22 \textit{ років} \tag{3.8}$$

Економічна ефективність E розраховується відповідно до формули:

$$E = \frac{E}{W} = \frac{290,08}{1226,02} = 0,23 \tag{3.9}$$

Основні економічні показники відображено у таблиці 3.4.

Таблиця 3.4 – Основні економічні показники

№ п/п	Показник	Обсяг
1	Сумарні витрати на розробку (W), грн	1226,02
2	Рентабельність,%	15
3	Ціна розробки, грн	1409,92
4	Плановий прибуток, грн	183,99
5	Термін окупності ($T_{ок}$), роки	4,22
6	Економічна ефективність, Е	0,23

Після проведення всіх розрахунків та аналізу аналогічних апаратних рішень було встановлено, що розробка дублікатора домофонних ключів, обійшлась дешевше, ніж аналогічні пристрої на сучасному ринку. Фактичний термін окупності витрат на реалізацію 4,22 років і коефіцієнт економічної ефективності 0,23 з нормативними 0,15 можна зробити висновок, що даний проект є економічно вигідним і доцільним.

ВИСНОВКИ

В кваліфікаційній роботі, на основі отриманих даних, було здійснено аналіз видів ключів, технологій ключів, домофонів, а саме: механічні ключі, які використовуються у повсякденному житті і у професіональних сферах; електронні ключі типу іммобілайзер і «Смарт ключ», які використовуються для захисту автомобілів та іншого транспорту; домофони та домофонні ключі, які в свою чергу захищають приміщення від не санкціонованого доступу;

Дослідження існуючих рішень для зчитування та копіювання Touch-memory та RFID міток дозволило здійснити опис технічних характеристик проектованого пристрою та зробити підбір елементної бази: програматор Arduino Nano, контактна луза для запису та зчитування інформації, кнопка перемикання режимів, резистори, підібрано джерело зовнішнього живлення.

Було здійснено побудову схеми підключення елементів системи керування та зроблено вихідний код для мікроконтролерної платформи, необхідний для функціонування пристрою.

Після розробки виконано тестування системи, а саме тестування зчитування, запису і симуляції домофонних ключів, визначено режими роботи, внесено необхідні виправлення.

Проведено розрахунок економічного ефекту та основних економічних показників, на основі яких можна зробити чіткий висновок про позитивну економічну доцільність створеного проекту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. «Смарт ключ» – URL: https://lester.ua/uk/discussions/sistema-beskljuchevogo-dostupa-v-avtomobile-dostoinstva-i-nedostatki (дата звернення 22.02.2023)
2. Комутатор – URL: https://ela.kpi.ua/bitstream/123456789/27992/1/OMPT_laboratorni.pdf (дата звернення 22.02.2023)
3. Система «Старт/Стоп» – URL: https://webshop-ua.intercars.eu/chitaite/News/yak-pratsiuie-systema-start-stop-i-navishcho-vona-potribna (дата звернення 22.02.2023)
4. Імобілайзер – URL: https://130.com.ua/uk/chto-takoe-immobilizer/ (дата звернення 26.02.2023)
5. Домофон – URL: https://www.telegroup.ua/ua/content/dom-systems/1/ (дата звернення 15.04.2023)
6. Технічне обслуговування домофонів – URL: http://www.dsz.com.ua/tekhnichne-obslugovuvannya.html (дата звернення 15.04.2023)
7. Дублікатор домофонних ключів – URL: https://newtravelers.ua/uk/oshibki/dublikator-domofonnyh-klyuchei-chto-eto-i-kak-ego-sdelat.html (дата звернення 15.04.2023)
8. Універсальні дублікаторидомофонних ключів – URL: https://rw1990.com.ua/product/universalnyy-dublikator-domofonnykh-klyuchey-tmd-5-proxi/ (дата звернення 15.04.2023)
9. Різновиди комп’ютерів – URL: https://bitkit.com.ua/rezistor (дата звернення 03.03.2023)
10. Покупка комплектуючих для проекту – URL: https://prom.ua/ua/Arduino-nano.html (дата звернення 06.05.2023)
11. Опис комплектуючі – URL: https://uk.wikipedia.org/wiki/ua (дата звернення 22.02.2023)
12. ArduinoIDE – URL: https://www.arduino.cc/en/software (дата звернення 22.02.2023)
13. Резистори – URL: https://radio-detali.com/ua/stati/rezistory-naznachenie (дата звернення 22.02.2023)
14. Енкодер – URL: https://stl-grupp.com/info/what-is-encoder.html (дата звернення 22.02.2023)
15. Крона – URL: https://dic.academic.ua/dic.nsf/ruwiki/618643 (дата звернення 22.02.2023)
16. Паяльна станція – URL: https://np.pl.ua/2020/08/shcho-take-paial-na-stantsiia/ (дата звернення 22.02.2023)

17. Термоусадка – URL: https://ital-tecno.com.ua/termousadka-abo-%D1%96zolyac%D1%96jna-str%D1%96chka-csho-vibrati (дата звернення 22.02.2023)

ДОДАТКИ

ДОДАТОК А

Вихідний код системи копіювання

```
#include <OneWire.h>
#include <OneWireSlave.h>
#include "pitches.h"
#include <EEPROM.h>
#include <OLED_I2C.h>
OLED myOLED(SDA, SCL); //створюємо екземпляр класу OLED з ім'ям myOLED
extern uint8_t SmallFont[];
extern uint8_t BigNumbers[];
#include "GyverEncoder.h"
#include "TimerOne.h"

//settings
#define rfidUsePWD 0        // ключ використовує пароль для зміни
#define rfidPWD 123456      // пароль для ключа
#define rfidBitRate 2       // Швидкість обміну з rfid до kbps
//pins
#define iButtonPin A3       // Лінія data ibutton
#define iBtnEmulPin A1      // Лінія эмулятора ibutton
#define Luse_Led 13         // Світлодіод лузи
#define R_Led 2             // RGB Led
#define G_Led 3
#define B_Led 4
#define ACpin 6             // Ain0 аналогового компаратора 0.1В для EM-Marie
#define speakerPin 12       // Спікер
#define FreqGen 11          // генератор 125 кГц
```

```
#define CLK 8              // s1 энкодера
#define DT 9               // s2 энкодера
#define BtnPin 10          // Кнопка перемикання режиму читання/запис

Encoder enc1(CLK, DT, BtnPin);
OneWire ibutton (iButtonPin);
OneWireSlave iBtnEmul(iBtnEmulPin);       //Емулятор iButton для BlueMode
byte maxKeyCount;                         // максимальна кількість ключів, що влазить в EEPROM, але не > 20
byte EEPROM_key_count;                    // кількість ключів 0..maxKeyCount, який зберігається в EEPROM
byte EEPROM_key_index = 0;                // 1..EEPROM_key_count номер останнього записаного в EEPROM ключа
byte addr[8];                             // тимчасовий буфер
byte keyID[8];                            // ID ключа для запису
byte rfidData[5];                         // значущі дані frid em-marine
byte halfT;                               // напівперіод для метаком
enum emRWType {rwUnknown, TM01, RW1990_1, RW1990_2, TM2004, T5557, EM4305};              // тип болванки
enum emkeyType {keyUnknown, keyDallas, keyTM2004, keyCyfral, keyMetacom, keyEM_Marine};    // тип оригінального ключа
emkeyType keyType;
enum emMode {md_empty, md_read, md_write, md_blueMode};              // режим роботи копіювальника
emMode copierMode = md_empty;

void OLED_printKey(byte buf[8], byte msgType = 0){
  String st;
  switch (msgType){
```

```
            case 0: st = "The key " + String(EEPROM_key_index) + " of " +
String(EEPROM_key_count) + " in ROM"; break;
            case 1: st = "Hold the Btn to save";  break;
            case 3: st = "The key " + String(indxKeyInROM(buf)) + " exists in ROM";
break;
          }
          myOLED.clrScr();
          myOLED.print(st, 0, 0);
          st = "";
          for (byte i = 0; i < 8; i++) st += String(buf[i], HEX) +":";
          myOLED.print(st, 0, 12);
          st = "Type ";
          switch (keyType){
            case keyDallas: st += "Dallas wire"; break;
            case keyCyfral: st += "Cyfral wire";  break;
            case keyMetacom: st += "Metakom wire"; break;
            case keyEM_Marine: st += "EM_Marine rfid"; break;
            case keyUnknown: st += "Unknown"; break;
          }
          myOLED.print(st, 0, 24);
          myOLED.update();
        }

        void OLED_printError(String st, bool err = true){
          myOLED.clrScr();
          if (err) myOLED.print(F("Error!"), 0, 0);
            else myOLED.print(F("OK"), 0, 0);
          myOLED.print(st, 0, 12);
          myOLED.update();
        }
```

```
void setup() {
  pinMode(Luse_Led, OUTPUT); digitalWrite(Luse_Led, HIGH); //замінити на LOW
  myOLED.begin(SSD1306_128X32); //ініціалізуємо дисплей
  pinMode(BtnPin, INPUT_PULLUP);                    // включаємо читання та підтягуємо пін кнопки режиму до +5В
  pinMode(speakerPin, OUTPUT);
  pinMode(ACpin, INPUT);                            // вхід аналогового компаратора 3В для Cyfral
  pinMode(R_Led, OUTPUT); pinMode(G_Led, OUTPUT); pinMode(B_Led, OUTPUT);  //RGB-led
  clearLed();
  pinMode(FreqGen, OUTPUT);
  Serial.begin(115200);
  myOLED.clrScr();                                  //Очищаємо буфер дисплея.
  myOLED.setFont(SmallFont);                        //Перед виведенням тексту необхідно вибрати шрифт
  myOLED.print(F("Hello, read a key..."), LEFT, 0);
  char st[16] = {98, 121, 32, 77, 69, 88, 65, 84, 80, 79, 72, 32, 68, 73, 89, 0};
  myOLED.print(st, LEFT, 24);
  myOLED.update();
  Sd_StartOK();
  EEPROM_key_count = EEPROM[0];
  maxKeyCount = EEPROM.length() / 8 – 1; if (maxKeyCount > 20) maxKeyCount = 20;
  if (EEPROM_key_count > maxKeyCount) EEPROM_key_count = 0;
  if (EEPROM_key_count != 0 ) {
    EEPROM_key_index = EEPROM[1];
    Serial.print(F("Read key code from EEPROM: "));
    EEPROM_get_key(EEPROM_key_index, keyID);
    for (byte i = 0; i < 8; i++) {
```

```
          Serial.print(keyID[i], HEX); Serial.print(F(":"));
        }
        Serial.println();
        delay(3000);
        OLED_printKey(keyID);
        copierMode = md_read;
        digitalWrite(G_Led, HIGH);
      } else {
        myOLED.print(F("ROM has no keys yet."), 0, 12);
        myOLED.update();
      }
      enc1.setTickMode(AUTO);
      enc1.setType(TYPE2);
      enc1.setDirection(REVERSE);         // NORM / REVERSE
      Timer1.initialize(1000);            // встановлення таймера на кожні 1000
мікросекунд (= 1 мс)
      Timer1.attachInterrupt(timerIsr);   // запуск таймера
      digitalWrite(Luse_Led, !digitalRead(Luse_Led));
    }

    void timerIsr() {   // переривання таймера для енкодера
      enc1.tick();
    }

    void clearLed(){
      digitalWrite(R_Led, LOW);
      digitalWrite(G_Led, LOW);
      digitalWrite(B_Led, LOW);
    }
```

```
byte indxKeyInROM(byte buf[]){ //повертає індекс або нуль якщо немає в ROM
  byte buf1[8]; bool eq = true;
  for (byte j = 1; j<=EEPROM_key_count; j++){  // шукаємо ключ в eeprom.
    EEPROM.get(j*sizeof(buf1), buf1);
    for (byte i = 0; i < 8; i++)
      if (buf1[i] != buf[i]) { eq = false; break;}
    if (eq) return j;
    eq = true;
  }
  return 0;
}

bool EPPROM_AddKey(byte buf[]){
  byte buf1[8]; byte indx;
  indx = indxKeyInROM(buf);                // шукаємо ключ в eeprom. Якщо знаходимо, то не робимо запис, а індекс переводимо до нього
  if ( indx != 0) {
    EEPROM_key_index = indx;
    EEPROM.update(1, EEPROM_key_index);
    return false;
  }
  if (EEPROM_key_count <= maxKeyCount) EEPROM_key_count++;
  if (EEPROM_key_count < maxKeyCount) EEPROM_key_index = EEPROM_key_count;
    else EEPROM_key_index++;
  if (EEPROM_key_index > EEPROM_key_count) EEPROM_key_index = 1;
  Serial.println(F("Adding to EEPROM"));
  for (byte i = 0; i < 8; i++) {
    buf1[i] = buf[i];
    Serial.print(buf[i], HEX); Serial.print(F(":"));
```

```
  }
  Serial.println();
  EEPROM.put(EEPROM_key_index*sizeof(buf1), buf1);
  EEPROM.update(0, EEPROM_key_count);
  EEPROM.update(1, EEPROM_key_index);
  return true;
}

void EEPROM_get_key(byte EEPROM_key_index1, byte buf[8]){
  byte buf1[8];
  int address = EEPROM_key_index1*sizeof(buf1);
  if (address > EEPROM.length()) return;
  EEPROM.get(address, buf1);
  for (byte i = 0; i < 8; i++) buf[i] = buf1[i];
  keyType = getKeyType(buf1);
}

emkeyType getKeyType(byte* buf){
  if (buf[0] == 0x01) return keyDallas;                    // це ключ формату dallas
  if ((buf[0] >> 4) == 0b0001) return keyCyfral;
  if ((buf[0] >> 4) == 0b0010) return keyMetacom;
  if ((buf[0] == 0xFF) && vertEvenCheck(buf)) return keyEM_Marine;
  return keyUnknown;
}

//***************** звуки***************
void Sd_ReadOK() {  // звук ОК
  for (int i=400; i<6000; i=i*1.5) { tone(speakerPin, i); delay(20); }
  noTone(speakerPin);
}
```

```
void Sd_WriteStep(){  // звук "черговий крок "
  for (int i=2500; i<6000; i=i*1.5) { tone(speakerPin, i); delay(10); }
  noTone(speakerPin);
}

void Sd_ErrorBeep() {  // звук "ERROR"
  for (int j=0; j <3; j++){
    for (int i=1000; i<2000; i=i*1.1) { tone(speakerPin, i); delay(10); }
    delay(50);
    for (int i=1000; i>500; i=i*1.9) { tone(speakerPin, i); delay(10); }
    delay(50);
  }
  noTone(speakerPin);
}

void Sd_StartOK(){   // звук "Успішне включення "
  tone(speakerPin, NOTE_A7); delay(100);
  tone(speakerPin, NOTE_G7); delay(100);
  tone(speakerPin, NOTE_E7); delay(100);
  tone(speakerPin, NOTE_C7); delay(100);
  tone(speakerPin, NOTE_D7); delay(100);
  tone(speakerPin, NOTE_B7); delay(100);
  tone(speakerPin, NOTE_F7); delay(100);
  tone(speakerPin, NOTE_C7); delay(100);
  noTone(speakerPin);
}
```