Хмельницький національний університет
Факультет інформаційних технологій
Кафедра комп'ютерної інженерії та інформаційних систем

КВАЛІФІКАЦІЙНА РОБОТА

бакалавр
Освітній рівень

Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino
Назва теми

КвРКІ 210363.21.03.66 ПЗ
Шифр

Галузь знань *12 «Інформаційні технології»*
Шифр, назва

Спеціальність *123 «Комп'ютерна інженерія»*
Шифр, назва

Освітня програма *«Комп'ютерна інженерія та програмування»*
Назва

Виконав: студент IV курсу, група КІ2-21-3 _____ Дмитрій ДАВИДОВ
Підпис Ініціали, прізвище

Керівник _____ Василь СТЕЦЮК
Підпис, дата Ініціали, прізвище

Нормоконтролер _____ Тетяна КИСІЛЬ
Підпис, дата Ініціали, прізвище

До захисту допускаю:
зав. кафедри комп'ютерної інженерії та інформаційних систем _____ Ольга ПАВЛОВА
Підпис Ініціали, прізвище

«18» червня 2025 р.

Хмельницький 2025

ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет Інформаційних технологій

Кафедра Комп'ютерної інженерії та інформаційних систем

Освітній рівень бакалавр

Галузь знань 12 Інформаційні технології

Спеціальність 123 Комп'ютерна інженерія

Освітня програма «Комп'ютерна інженерія та програмування»

ЗАТВЕРДЖУЮ

Зав. кафедри Ольга ПАВЛОВА

" 10 " 01 2025 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ БАКАЛАВРА

Дмитрію ДАВИДОВУ
Прізвище, ім'я, по батькові студента

1. Тема проекту (роботи) Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino

Керівник проекту (роботи) Василь СТЕЦЮК, ст. викладач
Прізвище, ім'я, по батькові, науковий ступінь, вчене звання

Затверджена наказом ректора університету від 07.02.2025 р. № 23

2. Строк подання студентом проекту (роботи) на кафедру 01.06.2025 р.

3. Вихідні дані до проекту (роботи) Завдання на кваліфікаційну роботу

4. Зміст пояснювальної записки (перелік питань, які потрібно розробити)

Аналіз відомих програматорів мікросхем пам'яті EPROM

Моделювання програматора мікросхем пам'яті EPROM для персонального комп'ютера

Програмно-апаратна реалізація програматора мікросхем пам'яті EPROM для персонального комп'ютера

5. Перелік графічного матеріалу (із зазначенням обов'язкових креслень)

Лістинг коду програми

Схема електрична принципова

Алгоритми роботи системи

6. Консультанти розділів дипломного проекту (роботи)

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Нормоконтроль	Тетяна КИСІЛЬ, доцент кафедри КПС		
Антиплагіат	Андрій НІЧЕПОРУК, доцент кафедри КПС		

7. Дата видачі завдання «_10_»_____01_____2025 р.

КАЛЕНДАРНИЙ ПЛАН

№з/п	Назва етапів (розділів) дипломного проекту (роботи)	Термін виконання етапів проекту (роботи)	Примітка
1	Вибір напряму дослідження та узгодження тематики кваліфікаційної роботи з керівником	10.01.2025	виконано
2	Ознайомлення з предметною областю; формулювання мети та задач дослідження; визначення об'єкта та предмета дослідження	01.02.2025	виконано
3	Робота над розділом 1 – дослідження предметної області та постановка задачі	01.03.2025	виконано
4	Робота над розділом 2 – вибір компонентів для програматора мікросхем пам'яті EPROM для персонального комп'ютера	01.04.2025	виконано
5	Робота над розділом 3 – проєктування програматора мікросхем пам'яті EPROM для персонального комп'ютера	29.04.2025	виконано
6	Оформлення пояснювальної записки згідно вимог	25.05.2025	виконано
7	Попередній захист ВКР	26.05.2025	виконано
8	Захист ВКР на засіданні ЕК	Червень 2025 року	

Студент _____ Дмитрій ДАВИДОВ
Підпис Ініціали, прізвище

Керівник роботи _____ Василь СТЕЦЮК
Підпис Ініціали, прізвище

№ рядка	формат	Позначення	Найменування	Кіл. листів	№ екз	Примітка
			Текстові документи			
1		КвРКІ 210363.21.03.66 ПЗ	Пояснювальна записка	60		
			Графічні матеріали			
2		КвРКІ 210363.21.03.66 Е8	Лістинг коду програми	1		
3		КвРКІ 210363.21.03.66 Е8	Схема електрична принципова	1		
4		КвРКІ 210363.21.03.66 Е8	Алгоритми роботи системи	1		

Зм	Арк	№ докум	Підпис	Дата	КвРКІ 210363.21.03.66 ВП		
Розробив		Давидов		06.6.25	Відомість проекту	Літера	Аркуш	Аркушів
Перевір.		Стецюк		12.6.25		У	1	1
Н. контр.		Кисіль				ХНУ, КІ2-21-3		
Затв.		Павлова		12.06.25				

АНОТАЦІЯ

Тема кваліфікаційної роботи: «Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino».

Автор роботи: Дмитрій ДАВИДОВ.

Керівник роботи: Стецюк Василь Микролайович.

Пояснювальна записка: 56 с., 18 рис., 4 табл., 3 дод., 50 джерел.

Графічна частина: 3 креслення.

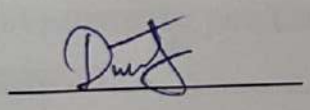
ARDUINO. ПРОГРАМАТОР, ПК, МІКРОСХЕМА, ПАМ'ЯТЬ, КОМП'ЮТЕР.

Об'єктом дослідження є мікросхеми постійної пам'яті EPROM та процеси їхнього електричного і програмного керування в умовах мікроконтролерної системи.

Предметом дослідження є технічні та програмні засоби реалізації програматора мікросхем EPROM із використанням платформи Arduino для забезпечення взаємодії з персональним комп'ютером.

Метою кваліфікаційної роботи є розробка та реалізація пристрою - програматора мікросхем пам'яті EPROM, що дозволяє здійснювати зчитування, запис і перевірку інформації з можливістю керування через персональний комп'ютер, із використанням апаратної платформи Arduino Mega 2560.

Під час проведення даного дослідженняя розроблена архітектура і компоненти програмного забезпечення для Arduino Mega 2560, що забезпечують повний цикл роботи з мікросхемами EPROM: ініціалізацію, генерацію адрес, подачу даних, формування імпульсу програмування, верифікацію результатів і обмін із ПК.



Підпис студента

30.05.2025

Дата

ЗМІСТ

ВСТУП 4

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПРИНЦИПИ ПРОГРАМУВАННЯ EPROM 6

1.1 Історія та класифікація мікросхем пам'яті 6

1.2 Принцип роботи мікросхем типу EPROM 8

1.3 Існуючі методи програмування EPROM 10

1.4 Аналіз готових рішень програматорів 13

1.5 Обґрунтування вибору платформи Arduino 16

1.6 Постановка задачі 19

1.7 Висновки до розділу 1 21

2 РОЗРОБКА ПРОГРАМАТОРА EPROM НА БАЗІ ПЛАТФОРМИ ARDUINO 24

2.1 Постановка задачі 24

2.2 Структурна та функціональна схема системи 29

2.3 Алгоритм функціонування пристрою 32

2.4 Розробка прошивки для Arduino 34

2.5 Висновки до розділу 2 39

3 ТЕСТУВАННЯ, НАЛАГОДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ 41

3.1 Методика тестування 41

3.2 Результати експериментів 43

3.3 Аналіз стабільності та ефективності роботи 47

3.4 Порівняльна характеристика з іншими рішеннями 49

3.5 Перспективи вдосконалення розробки 53

3.6 Висновки до розділу 3 56

ВИСНОВКИ 59

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ 61

ДОДАТОК А 66
ДОДАТОК Б 67
ДОДАТОК В 68

ВСТУП

На сучасному етапі розвитку техніки та цифрових технологій особливої актуальності набувають електронні пристрої, здатні забезпечити тривале зберігання інформації навіть у разі повного знеструмлення системи. Серед усього спектру засобів постійної пам’яті помітне місце займають мікросхеми типу EPROM (Erasable Programmable Read-Only Memory), які завдяки своїм конструктивним та функціональним особливостям продовжують залишатися у використанні навіть попри появу новіших рішень на кшталт EEPROM та Flash. Їх не поспішають повністю витіснити новітні технології, адже EPROM і досі демонструє стабільність, надійність та передбачувану поведінку в умовах, коли часте оновлення прошивки не є пріоритетом, а акцент зроблено на збереженні даних на тривалий термін.

Характерною відмінністю мікросхем EPROM є те, що інформація, записана в них, може зберігатися протягом багатьох років без втрати даних навіть у несприятливих умовах. У разі потреби змінити прошивку - стирання вмісту відбувається за допомогою ультрафіолетового випромінювання, що фізично очищає комірки пам’яті.

Водночас робота з такими мікросхемами вимагає використання спеціалізованих пристроїв - програматорів, які забезпечують можливість запису та зчитування інформації у відповідному електричному режимі. На практиці більшість комерційно доступних програматорів мають низку обмежень: складну конструкцію, обмежену документацію, високу вартість або закритий програмний інтерфейс. Такі фактори суттєво ускладнюють їх використання в навчальних цілях, у лабораторних умовах, для самостійних розробок, а також у випадках, коли проєкт розраховано на використання доступних ресурсів без надлишкових інвестицій у апаратне забезпечення.

Вирішенням цієї проблеми може стати створення власного, спрощеного за конструкцією програматора EPROM, який базуватиметься на відкритій та гнучкій

платформі. Ідеальним варіантом у такому випадку є використання Arduino - популярної мікроконтролерної платформи, що поєднує в собі легкість освоєння, широкий спектр готових бібліотек, підтримку багатьох периферійних пристроїв і активну спільноту, яка постійно ділиться новими розробками. Arduino Mega 2560, обраний для реалізації даного проєкту, забезпечує достатню кількість цифрових портів для адресації та передачі даних, а також можливість реалізації управління програмною напругою, необхідною для запису в EPROM.

Розроблений пристрій передбачає повноцінну взаємодію між персональним комп'ютером, мікроконтролером Arduino та мікросхемою пам'яті. Завдяки цьому користувач отримує можливість зручно записувати дані до пам'яті, зчитувати їх назад, контролювати процес програмування та автоматизувати робочий цикл. Особлива увага приділена розробці програмного інтерфейсу, що забезпечує простоту у використанні та розширюваність. Такий підхід надає змогу не тільки відтворити основні функції дорогих промислових пристроїв, але й адаптувати рішення до конкретних навчальних або дослідницьких потреб, що надзвичайно важливо у сфері технічної освіти та прикладної інженерії.

Крім того, створення власного програматора EPROM сприяє кращому розумінню внутрішніх процесів, що відбуваються при зчитуванні та записі даних. Процес реалізації пристрою передбачає роботу з принциповими схемами, опрацювання електричних характеристик, формування імпульсів керування, а також забезпечення стабільності подачі напруги програмування.

Основна мета проєкту полягає у створенні доступного, зрозумілого та ефективного пристрою для запису в мікросхеми EPROM, орієнтованого як на практичне використання, так і на ознайомлення з фундаментальними аспектами побудови цифрових систем пам'яті. Очікується, що запропоноване рішення стане корисним інструментом для студентів, інженерів-початківців та розробників, які працюють у сфері електроніки, мікроконтролерних технологій та автоматизованих систем керування.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ ТА ПРИНЦИПИ ПРОГРАМУВАННЯ EPROM

1.1 Історія та класифікація мікросхем пам’яті

Постійна пам’ять - це невід’ємна складова практично кожної цифрової системи, яка забезпечує збереження даних незалежно від джерела живлення. На відміну від оперативної пам’яті, яка втрачає свій вміст після вимкнення електроживлення, мікросхеми постійної пам’яті гарантують збереження інформації впродовж тривалого часу, навіть за відсутності живлення. Саме це робить їх особливо важливими для зберігання мікропрограм, налаштувань, прошивок і керувальних команд, які мають бути завантажені при кожному запуску системи [1].

З моменту появи перших мікропроцесорних пристроїв потреба у стабільному носії для збереження коду була надзвичайно великою [2]. Саме тому розвиток технологій постійної пам’яті йшов паралельно з прогресом у галузі мікроелектроніки. Найперші мікросхеми постійної пам’яті, відомі як маскові ROM (Mask ROM), виготовлялися з фіксованим вмістом, який «зашивався» безпосередньо під час виробництва мікросхеми на етапі фотолітографії [3]. У разі виявлення помилки або потреби у зміні логіки роботи системи, потрібно було повністю виготовляти нову партію мікросхем [4]. Це було вкрай затратним і неефективним, особливо у випадках малосерійного виробництва.

На зміну масковим ROM прийшли PROM (Programmable Read-Only Memory), які дозволяли одноразове програмування користувачем [5]. У цих мікросхемах запис даних здійснювався за допомогою спеціального високовольтного програматора, який «перепалював» мікроскопічні запобіжники в структурі чипа. Це відкривало нові можливості, однак усе ще не дозволяло внести зміни після першого запису. Тобто будь-яка помилка програмування чи необхідність у оновленні вимагала фізичної заміни мікросхеми на нову [6].

Кардинальною зміною стала поява EPROM - Erasable Programmable Read-Only Memory, або стираних програмованих ПЗП. Ці мікросхеми мали значно більшу гнучкість, адже дозволяли багаторазове програмування. Їх архітектура базувалася на використанні плаваючого затвора, який зберігає заряд після запису [7]. Стирання вмісту здійснювалося за допомогою ультрафіолетового випромінювання - для цього в корпусі мікросхеми розміщувалося кварцове віконце, через яке опромінювався кристал [8]. Цей процес тривав від 10 до 30 хвилин залежно від типу лампи та інтенсивності випромінювання. Після стирання усі біти набували значення логічної одиниці, що дозволяло здійснити новий цикл програмування [9].

Наявність можливості стирання та повторного запису зробила EPROM справжнім проривом у процесі розробки апаратного забезпечення. Розробники отримали змогу багаторазово налагоджувати та тестувати прошивки, не витрачаючи ресурси на виготовлення нових чипів після кожної зміни. Це радикально зменшило вартість експериментів, прискорило інженерні цикли та значно підвищило гнучкість процесу розробки електроніки [10].

Однак EPROM мали й свої обмеження. Процес стирання був повільним і потребував спеціального обладнання - ультрафіолетового стерилізатора з кварцовою лампою певної довжини хвилі (254 нм). Крім того, багаторазове опромінення поступово погіршувало якість мікросхеми, зменшуючи кількість гарантованих циклів перезапису [11].

Зі зростанням вимог до зручності з’явилися EEPROM (Electrically Erasable PROM) - мікросхеми, які можна стирати та переписувати електрично, тобто без будь-якого опромінення або демонтажу. EEPROM стали логічним продовженням розвитку технологій ПЗП, забезпечуючи ще вищий рівень інтеграції, зручності й швидкодії [12]. У таких мікросхемах стирається не весь вміст одразу, а окремі байти або блоки, що робить їх ідеальними для конфігураційних даних, журналів подій, налаштувань користувача тощо.

Найпоширенішим типом енергонезалежної пам’яті на сьогодні є Flash, яка є різновидом EEPROM, але оптимізована для зберігання великих обсягів інформації. Вона використовується в USB-накопичувачах, твердотільних накопичувачах, картках пам’яті та багатьох мікроконтролерах. Flash має обмежену кількість циклів запису, але достатню для більшості застосувань [13].

Попри наявність нових типів пам’яті, EPROM зберігає свою нішу. Її використовують у ретро-техніці, для підтримки сумісності з уже наявним обладнанням, у навчальних цілях, а також у проєктах, де потрібна висока стабільність і захист від випадкового перезапису [14]. Навіть сучасні інженери часто стикаються з необхідністю працювати з EPROM, особливо під час реставрації старих промислових контролерів або створення сумісних копій прошивок для техніки минулих десятиліть [15].

У підсумку, еволюція мікросхем постійної пам’яті пройшла довгий шлях: від незмінних ROM до ультрасучасних Flash-накопичувачів. Кожен етап цієї еволюції був важливим кроком у напрямі підвищення гнучкості, надійності та зручності використання пам’яті [16]. У рамках цього дипломного проєкту особливу увагу приділено EPROM - як важливому перехідному рішенню, що поєднує простоту реалізації, наочність апаратних принципів та високу стабільність збереження даних. Саме ці якості роблять мікросхеми EPROM ідеальним об’єктом дослідження та практичної реалізації програматора на базі відкритої платформи Arduino [17].

1.2 Принцип роботи мікросхем типу EPROM

Мікросхеми EPROM (Erasable Programmable Read-Only Memory) є різновидом постійної пам’яті, що дозволяє багаторазове програмування з можливістю стирання вмісту. Вони зберігають дані без живлення, що робить їх особливо корисними в системах, де важлива стабільність і збереження прошивки на тривалий період. Принцип дії цих мікросхем базується на фізичних властивостях метал-оксид-напівпровідникових структур, зокрема ефекті накопичення заряду в ізольованому затворі транзистора [18].

У кожному осередку пам’яті EPROM розміщується транзистор із плаваючим затвором, який ізольований шаром діелектрика. Такий транзистор має дві затворні області: керувальний і плаваючий затвори. Плаваючий затвор фізично ізольований від інших елементів за допомогою оксиду кремнію, що забезпечує його електричну відокремленість. Під час програмування на затвор подається високовольтний імпульс (зазвичай від 12 до 25 В), унаслідок чого електрони проходять через ізоляційний шар і накопичуються у плаваючому затворі [19]. Цей заряд змінює електричні характеристики транзистора, зокрема порогову напругу, що унеможливлює або дозволяє протікання струму через канал. Таким чином, транзистор переходить у стан, який інтерпретується як логічна «1» або «0» [20].

Процес програмування мікросхем EPROM відбувається шляхом подачі адресного сигналу, даних, а також програмуючої напруги. Зазвичай, це виконується послідовно - спочатку подається адреса комірки пам’яті, потім дані для запису, після чого активується програмувальний імпульс. Важливим аспектом є контроль тривалості програмуючого імпульсу, оскільки надмірна напруга або час впливу можуть пошкодити комірку. У промислових програматорах цей процес автоматизовано, тоді як в аматорських рішеннях, зокрема на базі Arduino, часто застосовуються спрощені таймінги, які обираються емпірично [21].

Стирається інформація з EPROM за допомогою ультрафіолетового випромінювання певної довжини хвилі (зазвичай 253,7 нм). Промінь проходить через спеціальне кварцове віконце в корпусі мікросхеми та поступово руйнує накопичений заряд у плаваючому затворі транзисторів [22]. Це призводить до повернення осередків пам’яті до початкового стану, який відповідає логічній «1». Процес стирання є одночасним для всієї мікросхеми, що є одним із недоліків у порівнянні з EEPROM або Flash, де можливо вибіркове стирання. Час впливу ультрафіолету залежить від інтенсивності випромінювання, конструкції лампи та розміщення мікросхеми у камері [23]. Як правило, для повного стирання EPROM необхідно 15-30 хвилин безперервного опромінення.

Читання даних із мікросхеми EPROM здійснюється без необхідності високих напруг або додаткових фізичних дій [24]. При подачі адреси на відповідні входи дешифратор вибирає потрібну комірку, і транзистор через відповідну логіку передає збережене логічне значення на вихідні контакти. Читання відбувається на стандартній робочій напрузі, зазвичай 5 В, і не призводить до зношування осередків пам’яті. Саме завдяки цій властивості EPROM можуть забезпечувати десятки тисяч циклів читання без втрати достовірності даних [25].

Ще однією перевагою мікросхем EPROM є довговічність зберігання інформації. У сприятливих умовах (стабільна температура, захист від ультрафіолету та електростатичних розрядів) дані можуть зберігатися до 10-20 років. Проте варто враховувати, що тривале зберігання за несприятливих умов може призводити до поступового витоку заряду, що спричиняє помилки при читанні.

Попри деякі обмеження, такі як необхідність повного стирання перед перепрограмуванням і відносна складність створення ультрафіолетового стирача, EPROM залишаються важливим інструментом у лабораторіях, навчальних закладах та серед радіоаматорів. Робота з ними дозволяє детально зрозуміти фізичні принципи зберігання даних, а також отримати практичні навички взаємодії з низькорівневими електронними компонентами [26].

1.3 Існуючі методи програмування EPROM

Програмування мікросхем EPROM відбувається шляхом зміни електричного стану їхніх осередків пам’яті за допомогою спеціалізованого пристрою - програматора. Такий процес потребує точної подачі програмуючих імпульсів високої напруги, а також суворого дотримання часових параметрів. Залежно від типу мікросхеми, способу взаємодії з нею та технічної реалізації, методи програмування EPROM можуть відрізнятися як за схемотехнікою, так і за програмною реалізацією [27].

Найбільш поширеним є класичний метод паралельного програмування, за якого всі адресні та інформаційні сигнали передаються безпосередньо на відповідні контакти мікросхеми. У цьому випадку програматор послідовно формує необхідні комбінації адрес і даних, подає відповідну високу напругу програмування на визначений контакт і витримує імпульс заданої тривалості [28]. Такий спосіб використовується як у комерційних промислових програматорах, так і в саморобних пристроях, зокрема на основі Arduino. Його основною перевагою є повний контроль над усіма процесами програмування, однак недоліком залишається необхідність роботи з великою кількістю ліній зв'язку, що потребує значної кількості виводів мікроконтролера або застосування допоміжних схем мультиплексування. Окрім того, такий метод вимагає точної синхронізації сигналів і врахування електричних характеристик мікросхем, що може створювати додаткові труднощі при проектуванні програматора.

Ще одним методом, який отримав розвиток із поширенням мікроконтролерних платформ, є побудова інтерфейсного програмування через буферизовані лінії або розширювачі портів. Наприклад, можуть використовуватись мікросхеми типу 74HC595 або спеціальні драйвери шин, які дозволяють зменшити кількість задіяних виводів, передаючи дані за допомогою серійної шини [29]. Такий підхід дозволяє реалізувати програматор з меншими апаратними вимогами, однак потребує чіткішого програмного керування і додаткового часу на обробку кожної операції. Перевагою є спрощення апаратної частини програматора, але недоліком - зменшення загальної швидкості програмування.

У більш професійних рішеннях використовуються автоматизовані програматори, які здатні визначати тип мікросхеми, контролювати рівні напруги, проводити перевірку записаних даних, стирати вміст EPROM перед новим записом, а також створювати докладні звіти про процес програмування [30]. Такі пристрої часто мають закриту апаратну і програмну частину та орієнтовані на виробниче використання. Проте їхня висока вартість робить їх малодоступними для широкого кола користувачів [31]. З технічного погляду, такі програматори реалізують повний

цикл взаємодії з мікросхемою - від ідентифікації типу до підтвердження успішності програмування, часто із підтримкою декількох сотень або навіть тисяч моделей мікросхем пам’яті.

Існують також програматори, які використовують універсальні інтерфейси підключення до комп’ютера - USB, UART або LPT, хоча останні поступово втрачають актуальність [32]. Взаємодія з програматором може здійснюватися через спеціальні програми, які дозволяють завантажити бінарний файл прошивки, задати параметри програмування та контролювати хід виконання операцій. У багатьох випадках програмне забезпечення супроводжується графічним інтерфейсом користувача, що спрощує роботу з пристроєм навіть для недосвідчених користувачів. Разом із тим, певна частина програматорів вимагає використання термінальних інтерфейсів або командного рядка, що є характерним для інженерних систем та середовищ розробки з відкритим кодом [33].

Вибір методу програмування EPROM залежить від кількох чинників: типу мікросхеми, доступного обладнання, вимог до швидкості та надійності процесу, а також цілей, які переслідує розробник. У рамках цієї роботи передбачається реалізація програматора з використанням мікроконтролера Arduino, що забезпечить достатній рівень контролю над сигналами та дозволить реалізувати основні етапи програмування в доступний і гнучкий спосіб [34].

Такий підхід дозволяє досягти балансу між простотою реалізації та функціональністю пристрою, зберігаючи при цьому розширюваність проєкту та можливість його вдосконалення в майбутньому. Крім того, використання Arduino як основи для побудови програматора відкриває широкі можливості для подальшої модернізації, зокрема впровадження функцій зчитування, перевірки записаних даних, створення графічного інтерфейсу для керування з комп’ютера або інтеграції із зовнішніми носіями інформації [35].

1.4 Аналіз готових рішень програматорів

На ринку представлено чимало готових рішень для програмування мікросхем EPROM, які можна умовно поділити на дві категорії: комерційні промислові пристрої та аматорські (саморобні) розробки, доступні широкому загалу. Кожна з цих категорій має свої особливості, переваги та недоліки, що обумовлює доцільність їх застосування в тих чи інших умовах [36].

Комерційні програматори, такі як Willem, MiniPro TL866, Batronix BX або Elnec BeeProg, вирізняються високою надійністю, широким переліком підтримуваних мікросхем та автоматизованими процесами програмування. Вони зазвичай мають зручне програмне забезпечення, яке дозволяє обирати тип мікросхеми, завантажувати прошивку, здійснювати перевірку запису, стирати дані, а також генерувати звіти. Більшість таких пристроїв підключаються до комп'ютера через USB-інтерфейс, мають компактний корпус і не потребують спеціального налаштування [37]. Наприклад, програматор TL866II Plus підтримує понад 15000 мікросхем, має швидкий режим запису та функцію перевірки цілісності даних (рис 1.1).

Рисунок 1.1 – TL866II Plus

Elnec BeeProg2, орієнтований на професійне використання, забезпечує високу точність та швидкість програмування, підтримує автоналаштування та забезпечує ізоляцію каналів для захисту мікросхем (рис 1.2).

Рисунок 1.2 – Elnec BeeProg2

У противагу комерційним рішенням існує велика кількість аматорських програматорів, побудованих на основі мікроконтролерів Arduino, AVR, PIC або Raspberry Pi.

Такі розробки часто поширюються як відкриті проєкти, що дозволяє користувачам модифікувати як апаратну, так і програмну частину під власні потреби [38]. Наприклад, у відкритому доступі можна знайти проєкти, які використовують Arduino Mega або Arduino Uno разом з дешифраторами адрес і буферами даних для взаємодії з мікросхемами типу 27C256 або 27C512.

Також існують повноцінні open-source системи на базі Arduino Nano з підтримкою OLED-дисплея та SD-карти для завантаження прошивок без підключення до ПК. Суттєвою перевагою таких пристроїв є їхня низька вартість, простота повторення, а також навчальна цінність для розробників-початківців.

Проте аматорські рішення часто мають обмежений функціонал, зокрема не завжди передбачено автоматичне стирання або перевірку запису, іноді відсутня

підтримка різних типів мікросхем, а процес програмування може бути тривалішим і менш стабільним [39].

До того ж, у більшості таких пристроїв відсутній захист від неправильного підключення або перевантаження, що підвищує ризик виходу з ладу як самого програматора, так і мікросхеми пам’яті.

Також слід згадати про гібридні рішення, які поєднують доступність відкритих платформ із функціональністю готових пристроїв.

Це, наприклад, програматори на базі Arduino з використанням зовнішніх модулів живлення, дисплеїв або навіть сенсорного інтерфейсу для взаємодії без ПК. У таких проєктах реалізуються покращені алгоритми запису й перевірки, що дозволяє наблизити їхню ефективність до комерційних аналогів при збереженні низької собівартості [40].

Для наочності нижче наведено порівняльну таблицю переваг та недоліків основних типів програматорів:

Таблиця 1.1 – Переваги та недоліки існуючих рішень

Тип програматора	Переваги	Недоліки
Комерційні (TL866, BeeProg)	Висока точність, широкий спектр підтримки, простота у використанні	Висока вартість, закритість апаратної частини
Аматорські (Arduino, AVR)	Низька ціна, гнучкість, навчальна користь, відкритий код	Обмежена підтримка мікросхем, складність розробки, нижча стабільність
Гібридні рішення	Баланс ціни та функцій, модульність, розширюваність	Потребують додаткових налаштувань, можуть мати складну структуру

Таким чином, аналіз готових рішень демонструє широкий спектр варіантів, які відрізняються за ціною, функціональністю, складністю реалізації та доступністю. Обираючи між ними, слід орієнтуватися на конкретні цілі: для навчання й експериментів цілком достатньо саморобних пристроїв, тоді як для масового виробництва або критичних застосувань доцільно обрати професійний комерційний програматор.

1.5 Обґрунтування вибору платформи Arduino

Під час розробки програматора мікросхем EPROM особливе значення має етап вибору апаратної платформи, яка виступає як серцевина всієї системи, визначаючи її можливості, гнучкість, надійність та технічний потенціал. Саме обрана платформа зумовлює спосіб реалізації комунікації з мікросхемами, здатність точно генерувати керувальні сигнали, обробляти введення та виведення даних, забезпечувати взаємодію з користувачем і програмним забезпеченням, а також підтримувати надійну роботу всіх підсистем пристрою у реальному часі.

Серед широкого переліку альтернатив, доступних для побудови вбудованих систем - таких як мікроконтролери AVR у низькорівневому варіанті, мікросхеми PIC від Microchip, 32-бітні контролери серії STM32, одноплатні комп’ютери типу Raspberry Pi, мікропроцесори ESP32 з підтримкою бездротових протоколів - вибір зупинено саме на Arduino, зокрема на моделі Arduino Mega 2560, яка найбільш повно відповідала вимогам до побудови надійного, функціонального та доступного програматора для мікросхем EPROM [41].

Arduino - це відкрита платформа, яка не лише надає фізичну апаратну основу у вигляді зручної для інтеграції плати, але й забезпечує розвинене програмне середовище (Arduino IDE), широке ком’юніті підтримки, численні готові бібліотеки та практично необмежений набір додаткових модулів. Основу цієї платформи становить мікроконтролер ATmega2560, який має 54 цифрові входи/виходи, з них 15 - із підтримкою PWM, 16 аналогових входів, а також чотири апаратні UART, що значно розширює можливості для керування периферією [42].

Наявність великої кількості цифрових ліній вводу/виводу є надзвичайно важливою для реалізації логіки програмування EPROM, яка передбачає одночасну роботу з 15-бітною адресною шиною, 8-бітною шиною даних та кількома сигналами керування - такими як OE (Output Enable), CE (Chip Enable), PGM (Program) та іншими. Arduino Mega 2560 дозволяє реалізувати усі ці сигнали без потреби в складних мультиплексорах або комбінаційній логіці, що суттєво спрощує конструкцію, прискорює розробку і зменшує ризик помилок, пов'язаних із затримками або конфліктами сигналів [43].

Ще одним критично важливим фактором є простота програмування платформи Arduino. Завдяки використанню Arduino IDE, процес написання, компіляції та завантаження прошивки відбувається практично без зайвих налаштувань. Розробник має можливість зосередитися виключно на реалізації логіки роботи програматора, не витрачаючи час на конфігурацію середовища, компіляторів або драйверів. Простота синтаксису, наближеного до мови C/C++, а також наявність великої кількості прикладів, шаблонів і відкритого коду дозволяють швидко інтегрувати будь-який функціонал - від формування імпульсів до обробки команд, що надходять із ПК через USB.

Ще однією незаперечною перевагою Arduino є її масштабованість та модульність [44]. Для реалізації зручного інтерфейсу з користувачем у розробці використовуються стандартні модулі - зокрема LCD-дисплей формату 1602, який підключається через інтерфейс I2C та дозволяє виводити поточну адресу, тип виконуваної операції, повідомлення про помилки або завершення циклу. Також до плати Arduino підключено зовнішні буфери (наприклад, 74LS245), регістри зсуву (74HC595), дешифратори (74HC138) та інші компоненти, взаємодія з якими реалізується без зайвих труднощів, оскільки Arduino підтримує керування як одиничними лініями, так і паралельними групами сигналів.

Важливо й те, що Arduino забезпечує надійну взаємодію з персональним комп'ютером через інтерфейс USB, який вбудовано у плату за замовчуванням. Arduino Mega 2560 має мікросхему CH340 або ATmega16U2 для реалізації USB-to-

Serial перетворення, що дозволяє підключати плату без необхідності в додаткових конвертерах або адаптерах45. У проєкті це забезпечує надзвичайно просту інтеграцію з комп'ютером для передачі команд (WRITE, READ, VERIFY), отримання відповідей, моніторингу поточного стану тощо.

Значну роль відіграє і вартісний аспект. Arduino Mega 2560, навіть у версії оригінального виробництва, залишається доступною у фінансовому плані. У поєднанні з дешевими компонентами та відкритими схемами - це робить платформу надзвичайно привабливою для освітніх проєктів, самостійного конструювання або малосерійного виробництва. Придбання плати, а також усіх сумісних периферійних модулів, не становить жодної складності, оскільки Arduino має величезний ринок, що охоплює як офіційні, так і сумісні з нею рішення [46].

Крім технічних параметрів, Arduino пропонує надзвичайно високий рівень підтримки з боку спільноти [47]. Сотні тисяч користувачів по всьому світу створили гігантську базу знань, документації, відеоуроків, покрокових інструкцій, проєктів на GitHub та тематичних форумів. Це дозволяє не лише швидко вирішувати технічні проблеми, але й активно вдосконалювати власну конструкцію, беручи за основу найкращі практики. У рамках реалізації цього програматора кілька важливих програмних рішень було адаптовано або створено на основі відкритих прикладів, що дозволило суттєво скоротити час розробки та зосередитися на тестуванні стабільності й зручності експлуатації.

Загалом платформа Arduino, зокрема Arduino Mega 2560, надає оптимальний баланс між функціональністю, доступністю, простотою реалізації, сумісністю з периферією та зручністю інтеграції в освітній або аматорський проєкт [48]. Завдяки їй вдалося реалізувати повноцінний програматор мікросхем EPROM, який поєднує точність формування таймінгів, гнучкість програмної логіки, стабільну комунікацію з комп'ютером та зручний інтерфейс із користувачем. Усі ці характеристики підтверджують, що вибір Arduino був не лише доцільним, але й стратегічно виваженим, з погляду технічного ресурсу, економічної ефективності та майбутньої масштабованості системи [49].

1.6 Постановка задачі

У межах реалізації даного проєкту було сформульовано комплексну задачу, що полягає у створенні ефективного, надійного та доступного програматора мікросхем EPROM, орієнтованого на використання як у навчальних, аматорських, так і в сервісних умовах. Особливий акцент зроблено на побудові системи, яка не тільки забезпечує виконання базових операцій запису та зчитування інформації, а й здатна автономно керувати процесами верифікації, сигналізації помилок, індикації стану та взаємодії з користувачем.

На початковому етапі перед розробником стояло завдання здійснити поглиблений аналіз архітектури та принципу дії мікросхем пам’яті типу EPROM серії 27Cxxx, які ще досі застосовуються в різних сферах - від промислового обладнання до ретро-комп’ютерів. Було вивчено методики електричного запису даних у такі мікросхеми, а також особливості їх стирання за допомогою ультрафіолетового випромінювання. Окрему увагу приділено схемотехнічним нюансам: рівням логічних сигналів, необхідним для активації режиму програмування, часовим інтервалам між подачею сигналів та високовольтної напруги, обмеженням щодо тривалості імпульсів, імовірності деградації комірок пам’яті тощо. Такий аналіз дозволив сформувати точне технічне бачення вимог до майбутнього пристрою.

Наступною задачею стало вивчення вже існуючих рішень у цій галузі. Проаналізовано як промислові програматори з широким функціоналом, так і саморобні проєкти, представлені ентузіастами в онлайн-спільнотах. Встановлено, що більшість комерційних пристроїв мають високу вартість, вимагають складного програмного забезпечення для ПК, або ж обмежені в підтримці типів мікросхем. У свою чергу, саморобні рішення часто страждають на нестачу стабільності, погану ергономіку чи відсутність належної індикації. Це дало змогу обґрунтувати необхідність створення власного пристрою з нуля, із врахуванням усіх виявлених

недоліків і з використанням гнучких, відкритих технологій, зокрема - платформи Arduino.

На основі проведеного аналізу сформульовано перелік функціональних вимог до системи. Пристрій повинен підтримувати роботу з мікросхемами EPROM, які мають 15-бітову адресну шину (що охоплює 32КБ адресного простору) та 8-бітову інформаційну шину. Потрібно забезпечити подачу програмної напруги рівнем 12.5 В із достатньою стабільністю та захистом від перенапруги. Програматор має забезпечувати повноцінний цикл операцій: генерацію адреси, передачу байта даних, подачу контрольних сигналів OE, CE, PGM, генерацію імпульсу запису, зчитування з пам’яті та автоматичну верифікацію записаного значення. У випадку розбіжності між записаним і зчитаним значенням система повинна реалізовувати повтор запису з подальшою перевіркою або повідомляти користувача про помилку.

Окрему задачу становила побудова архітектури пристрою. Враховуючи необхідність одночасного керування щонайменше 28 цифровими лініями (15 адресних, 8 для даних, 5 для керування та допоміжних функцій), було обґрунтовано вибір Arduino Mega 2560 - єдиної моделі з необхідною кількістю виводів і стабільною роботою через USB. Для розширення можливостей керування шинами адрес і даних вирішено залучити допоміжні мікросхеми: 74HC595 як регістр зсуву для адреси, 74LS245 як буфер між Arduino і шиною даних, а також дешифратори 74HC138 для формування контрольних сигналів. Крім того, для генерації високовольтного імпульсу обрано DC-DC перетворювач XL6009 з можливістю точного налаштування вихідної напруги.

Завданням програмного рівня стало розроблення прошивки, яка здатна у режимі реального часу формувати необхідні сигнали, дотримуючись мікросекундних затримок між фазами. Прошивка повинна мати зручну логіку обробки команд, які надходять з ПК через USB, бути достатньо гнучкою для адаптації під інші типи мікросхем і мати мінімальну затримку у виконанні критичних операцій.

Іншим критичним завданням було забезпечення індикації та зворотного зв’язку. Для цього передбачено LCD-дисплей формату 1602, на якому має відображатися статус операції, поточна адреса, тип команди (WRITE, READ, VERIFY), а також повідомлення про помилки або завершення процесу. Крім дисплея, пристрій має оснащуватись світлодіодами для візуалізації стану живлення, активності процесу програмування та наявності помилки.

Особливу увагу приділено безпеці пристрою. У межах постановки задачі визначено необхідність реалізації програмного захисту від некоректної послідовності дій: наприклад, блокування подачі програмної напруги при відсутності підтвердження наявності мікросхеми або відсутності даних у буфері. Також слід передбачити можливість зупинки циклу програмування в будь-який момент через втручання користувача або в разі виявлення нестабільної напруги.

Результатом виконання всього комплексу поставлених задач має стати створення повнофункціонального пристрою, що працює стабільно, без збоїв, забезпечує точне виконання операцій програмування, підтримує зворотний зв’язок, та є відтворюваним для інших розробників. Готовий прототип має забезпечити підтвердження працездатності розроблених схем, ефективність алгоритмів управління та відповідність усім висунутим вимогам.

У підсумку сформульоване завдання охоплює весь цикл інженерного проєктування - від аналізу технічної проблематики й формування вимог до створення апаратної та програмної частини, її тестування, налагодження та перевірки на реальних мікросхемах. Повна реалізація цього завдання є підтвердженням того, що обрана стратегія проєктування відповідає актуальним технічним і практичним потребам.

1.7 Висновки до розділу 1

У межах першого розділу дипломного проєкту здійснено комплексне теоретичне опрацювання предметної області, яка охоплює ключові аспекти

побудови, функціонування та програмування мікросхем постійної пам’яті типу EPROM.

На основі зібраної інформації сформовано цілісне уявлення про історичну еволюцію технологій енергонезалежного зберігання даних - від перших масочних ROM до багаторазово програмованих EPROM, з подальшим переходом до більш досконалих технологій EEPROM та Flash. Такий історичний контекст дозволив чітко окреслити місце мікросхем EPROM у загальній класифікації та зрозуміти їхню поточну нішу в сучасних електронних пристроях.

Особливу увагу приділено вивченню мікросхем серії 27C256, що є репрезентативним прикладом класичної EPROM. Проведено ґрунтовний аналіз їхньої внутрішньої структури, принципів адресації осередків, формування логічних рівнів на виводах, а також електричних і часових параметрів, необхідних для коректного програмування та зчитування даних.

Детально розглянуто особливості роботи з адресними та інформаційними шинами, специфіку генерації керувальних сигналів (OE, CE, PGM) та вимоги до стабільності програмної напруги.

Отримані знання лягли в основу розуміння того, як саме реалізується процес запису даних в EPROM, і які умови необхідно забезпечити для успішного функціонування мікросхеми.

Крім аналізу самої мікросхеми, вивчено існуючі методики її програмування, серед яких особливе місце займають паралельні програматори. Саме вони дозволяють безпосередньо керувати кожною шиною мікросхеми, що забезпечує високу точність і контроль над усім процесом.

При цьому встановлено, що велика частина промислових програматорів, наявних на ринку, є складними у конструкції, потребують спеціалізованого програмного забезпечення, мають закриту архітектуру або недружній інтерфейс. Такий стан справ суттєво ускладнює їх використання для навчання, прототипування або роботи в умовах обмежених ресурсів. Через це виникає

обґрунтована потреба у створенні власного програматора, побудованого на принципах відкритості, простоти та доступності.

Результати аналізу готових апаратних рішень дозволили виокремити низку конструктивних схем, що демонструють ефективні підходи до побудови EPROM-програматорів. До таких підходів належать використання регістрів зсуву для оптимізації кількості виводів, буферів для підвищення надійності передавання сигналів, дешифраторів для адресації та генераторів високої напруги, необхідної для програмування.

Ці напрацювання слугували базою для створення власної концепції апаратної реалізації, яка має бути адаптованою до обраної мікроконтролерної платформи.

Окремо розглянуто платформу Arduino, зокрема модель Arduino Mega 2560, яка обрана як базова для реалізації програматора.

На тлі порівняння з іншими мікроконтролерними платами саме ця модель виявилася найбільш придатною для даної задачі завдяки великій кількості цифрових виводів, що дозволяє одночасно керувати всіма необхідними лініями адреси та даних.

Загалом теоретична база, сформована у першому розділі, створює міцне підґрунтя для переходу до етапу практичної реалізації проєкту.

У результаті виконаної роботи сформовано чітке розуміння того, як повинна виглядати архітектура програматора, які елементи необхідно включити до схемотехнічної частини, які сигнали слід формувати для керування мікросхемою, а також які функції має реалізовувати програмна частина на платформі Arduino.

Усе це дозволяє з упевненістю перейти до наступного етапу, в межах якого буде здійснено побудову апаратної схеми, написання прошивки, проведення експериментального тестування та формування висновків щодо ефективності створеного пристрою.

2 РОЗРОБКА ПРОГРАМАТОРА EPROM НА БАЗІ ПЛАТФОРМИ ARDUINO

2.1 Постановка задачі

Розробка програматора мікросхем EPROM на базі платформи Arduino передбачає створення доступного, функціонального й надійного пристрою, здатного здійснювати операції запису, стирання та зчитування даних із мікросхем типу 27Cxxx. Основною метою є забезпечення користувачеві можливості працювати з мікросхемами EPROM без використання дорогого комерційного обладнання, шляхом реалізації недорогої альтернативи з використанням широко доступної апаратної бази.

Поставлене завдання передбачає реалізацію як апаратної, так і програмної частини системи. Зокрема, необхідно спроєктувати електричну схему програматора, підібрати відповідні компоненти для адресації, передачі даних і подачі напруги програмування, а також забезпечити живлення мікросхеми у відповідних режимах. З програмної точки зору, потрібно реалізувати алгоритм послідовного програмування осередків пам’яті, перевірки коректності запису та підтримки інтерфейсу взаємодії з користувачем, наприклад, через підключення до ПК або відображення інформації на дисплеї.

У процесі реалізації проєкту передбачається використання таких основних апаратних компонентів:

- плата Arduino Mega 2560 (рис 2.1) - мікроконтролер з великою кількістю цифрових виводів, достатньою пам’яттю для реалізації програмної логіки та підтримкою підключення до ПК через USB;

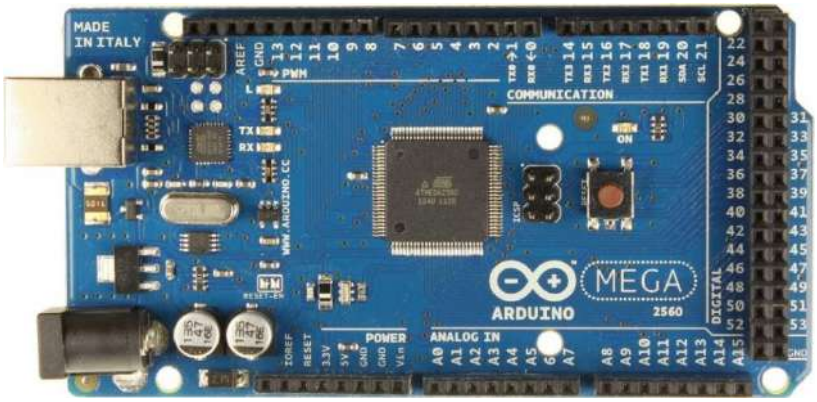


Рисунок 2.1 – Arduino Mega 2560

– EPROM мікросхема типу 27C256 (рис 2.2) - як базовий елемент для тестування програматора;

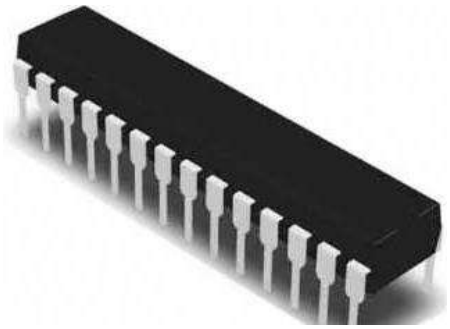


Рисунок 2.2 - EPROM мікросхема типу 27C256

– мікросхеми дешифраторів (74HC138) (рис 2.3) - для формування адресних сигналів;

Рисунок 2.3 - Мікросхема дешифраторів 74HC138

– регістри зсуву або буфери (74HC595, 74LS245) (рис 2.4) - для виведення даних та забезпечення логічного узгодження сигналів;

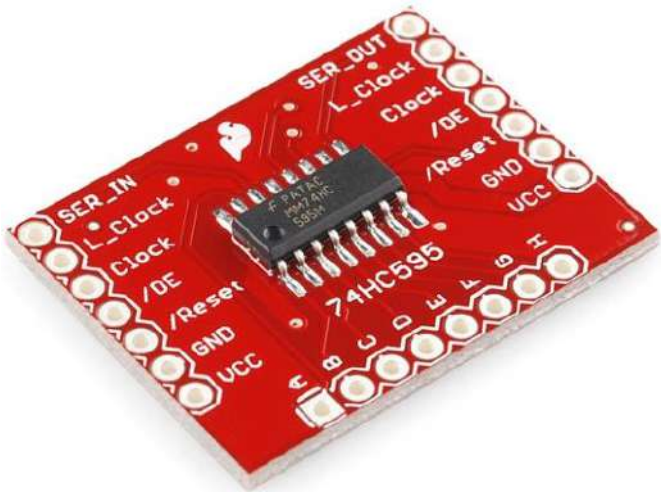


Рисунок 2.4 – Модуль з регістром зсуву 74HC595

– DC-DC конвертер XL6009 (рис 2.5) - для формування програмної напруги в діапазоні 12-21 В;

Рисунок 2.5 – Модуль XL6009

– LCD-дисплей 1602 (рис 2.6) - для виведення інформації про процес роботи;

Рисунок 2.6 – LCD-дисплей 1602

– світлодіоди, кнопки, резистори, макетна плата, провідники - як допоміжні елементи інтерфейсу та управління;

– ультрафіолетовий ліхтар (рис 2.7) - для стирання вмісту EPROM (зовнішній пристрій).

Рисунок 2.7 – Ультрафіолетовий стерилізатор з кварцовою лампою UVC 254

Особливу увагу слід приділити безпечному формуванню високовольтних імпульсів, які необхідні для запису в EPROM, а також синхронізації всіх сигнальних ліній між Arduino і програмованою мікросхемою. Важливо також забезпечити можливість стирання мікросхем (зокрема, передбачити умови для роботи із зовнішнім ультрафіолетовим стирачем) і повторного використання пристрою без потреби ручної перенастройки.

У межах розробки необхідно врахувати специфіку роботи мікросхем EPROM, які потребують подачі програмної напруги значно вищої, ніж стандартні рівні логічних сигналів Arduino. Це зумовлює необхідність у додаткових модулях або схемах, здатних формувати стабільне джерело напруги 12-21 В (залежно від моделі EPROM) з відповідною швидкодією. Крім того, потрібно врахувати обмеження по струму, який споживає мікросхема в момент запису - ці значення не повинні перевищувати допустимі межі для електронних компонентів програматора.

З урахуванням широкого спектра існуючих мікросхем EPROM, проєкт повинен мати можливість масштабування - зокрема, можливість адаптації до роботи з мікросхемами різного обсягу пам’яті та форм-фактору. Це потребує реалізації універсального підходу до адресації, з урахуванням варіативності кількості адресних ліній та структури внутрішньої організації пам’яті.

До обов’язкових функціональних можливостей розроблюваного пристрою слід також віднести перевірку правильності програмування (після кожного байта або блоку), можливість стирання вмісту (за підтримки ультрафіолетового джерела), наявність ручного або автоматичного режиму роботи, а також відображення поточного статусу операцій. Для цього доцільним є включення до складу пристрою елементів індикації (наприклад, світлодіодів або дисплея), а також реалізація зручного інтерфейсу обміну з ПК через USB.

Проєкт має відповідати низці критеріїв, зокрема: доступність компонентної бази, простота повторення, розширюваність, мінімальні вимоги до зовнішніх

ресурсів, а також зручність у використанні для кінцевого користувача. Крім того, важливою є модульність системи - її можливість адаптуватися до різних типів мікросхем або розширити функціонал у майбутньому.

У результаті реалізації поставлених задач буде створено програматор, що дозволяє проводити повноцінну роботу з мікросхемами EPROM, включаючи запис, зчитування та попередню перевірку, із можливістю взаємодії з комп'ютером або іншими зовнішніми пристроями, з потенціалом розширення функціональності під нові вимоги чи стандарти.

2.2 Структурна та функціональна схема системи

Для повноцінного розуміння роботи програматора EPROM доцільно розглянути його з позиції функціональної архітектури, яка включає основні логічні та апаратні підсистеми. Основу програматора становить плата Arduino Mega 2560, яка виконує роль центрального керуючого елементу. Вона координує взаємодію всіх компонентів системи, формує цифрові сигнали, керує послідовністю операцій, а також забезпечує обмін інформацією з персональним комп'ютером через USB-інтерфейс.

Адресація в системі реалізується за допомогою дешифраторів типу 74HC138. Їх використання дозволяє зменшити кількість задіяних виводів мікроконтролера, зберігаючи при цьому можливість повного охоплення адресного простору мікросхеми EPROM. Це рішення оптимізує структуру проєкту та підвищує його гнучкість у разі масштабування.

Передача даних між контролером і мікросхемою пам'яті здійснюється через регістри зсуву (74HC595) та буфери (74LS245), що забезпечує необхідне електричне узгодження сигналів і дозволяє реалізувати як односторонній, так і двонаправлений зв'язок. Такий підхід надає змогу виконувати як операції запису, так і зчитування інформації з мікросхеми, забезпечуючи при цьому надійність і швидкодію обміну (рис 2.8).

Для реалізації програмної напруги, необхідної під час запису в EPROM, у системі передбачено блок підвищення напруги на основі стабілізованого DC-DC перетворювача XL6009. Управління цим блоком здійснюється мікроконтролером шляхом подачі керуючого сигналу на електронний ключ, що забезпечує контрольований і безпечний процес подачі високої напруги.

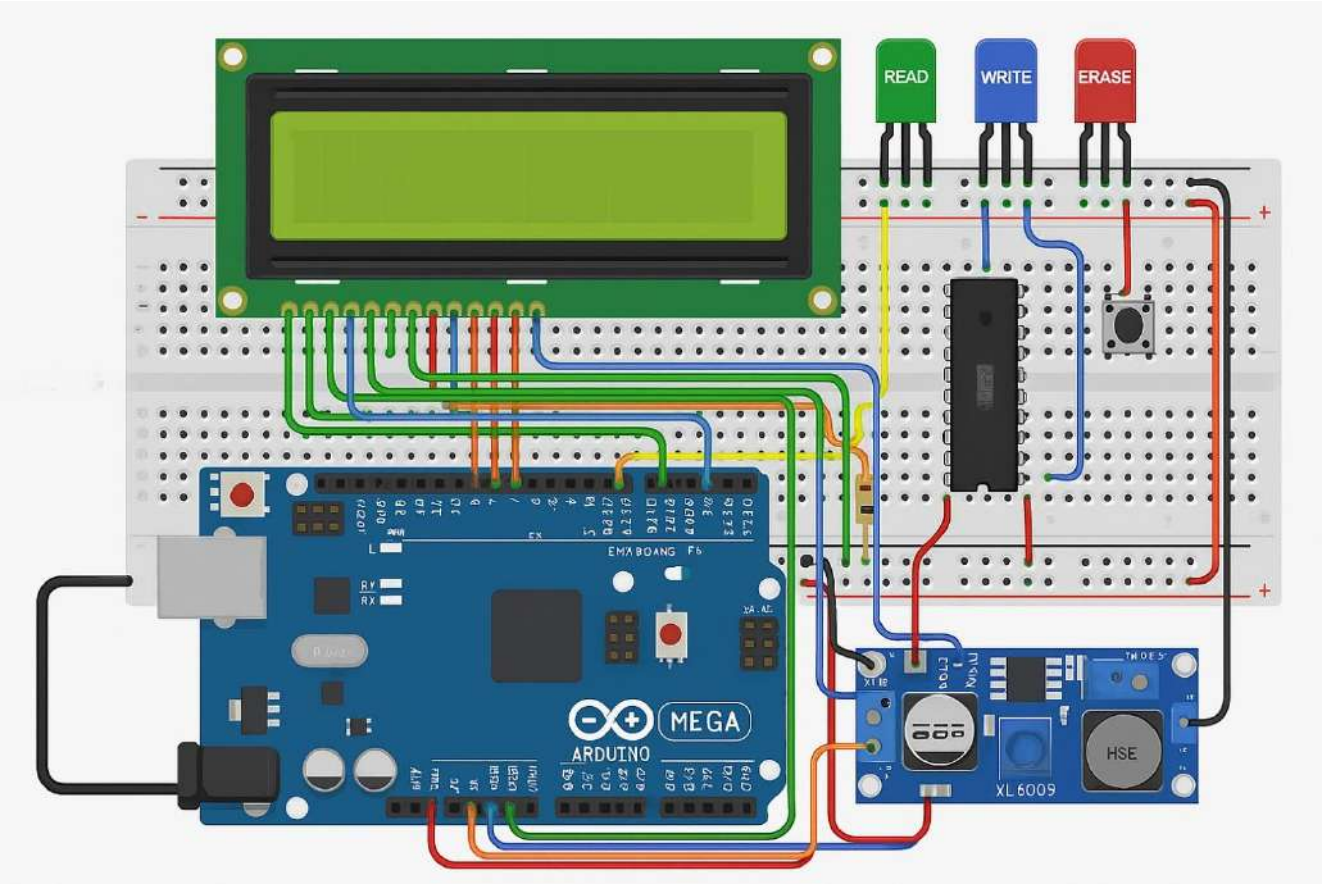


Рисунок 2.8 – Схема монтажна

Виведення інформації про стан роботи пристрою реалізовано за допомогою LCD-дисплея типу 1602. На ньому можуть відображатися повідомлення про активний режим, стан програмування, наявність помилок тощо. Допоміжні елементи, зокрема світлодіоди та кнопки, слугують простим засобом ручного керування і відображення статусу виконання окремих операцій.

Для стирання вмісту мікросхем EPROM використовується зовнішній ультрафіолетовий стерилізатор на основі кварцової лампи з довжиною хвилі 254

нм. Цей пристрій забезпечує надійне очищення пам’яті перед повторним записом, хоча фізично не інтегрується в основну електронну схему програматора. Його використання є обов’язковим етапом у повному циклі роботи з мікросхемами даного типу (рис 2.9).

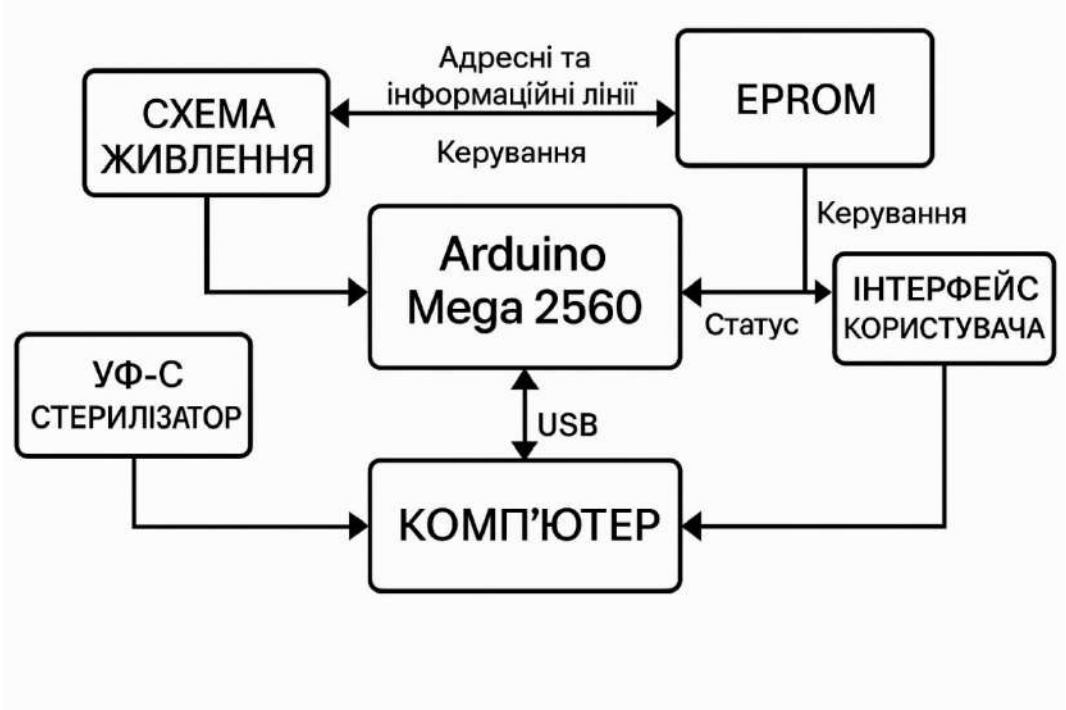


Рисунок 2.9 – Функціональна схема

Функціонально система працює за наступним принципом: мікроконтролер послідовно генерує адреси, формує дані, керує подачею програмної напруги та контролює результати запису шляхом зчитування і верифікації. Отримані результати можуть передаватися на комп’ютер для подальшого аналізу або зберігатися у внутрішній пам’яті Arduino. Така архітектура дозволяє реалізувати повноцінний програматор, придатний для експлуатації як у лабораторних умовах, так і в аматорських середовищах.

Отже, структурна й функціональна побудова програматора базується на тісній взаємодії між доступними логічними модулями (рис 2.10), які разом забезпечують ефективне й надійне виконання всіх необхідних функцій - від адресації та передачі даних до керування високовольтними режимами й виводу

інформації про стан системи. Такий підхід забезпечує універсальність, доступність і зручність розробленого пристрою.

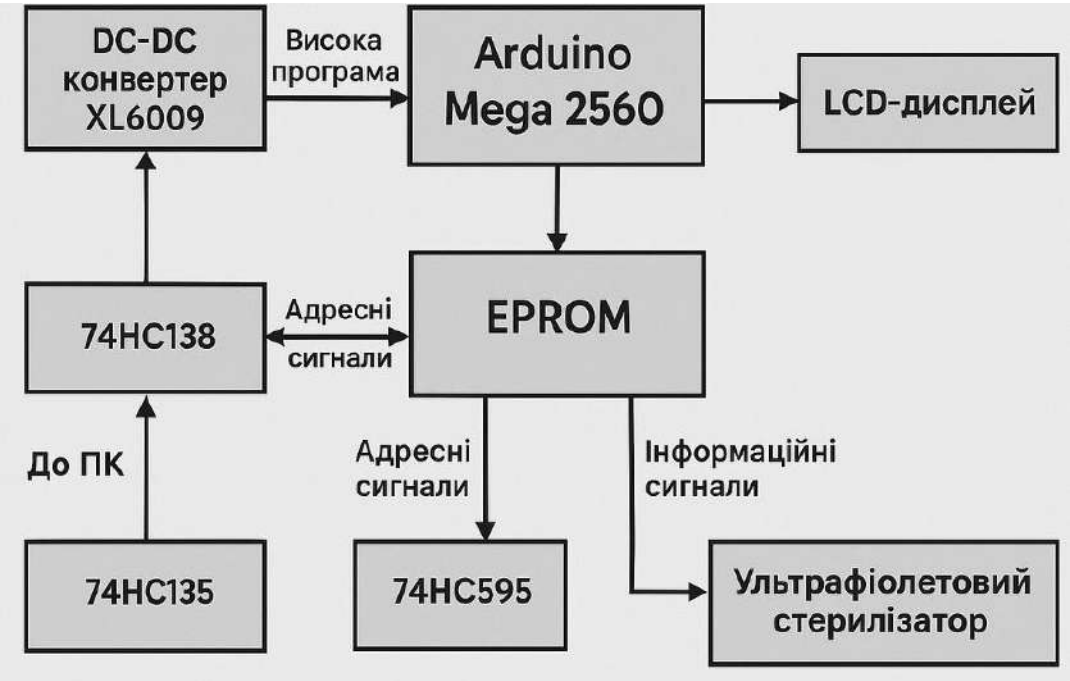


Рисунок 2.10 – Структурна схема

2.3 Алгоритм функціонування пристрою

Функціонування програматора EPROM базується на послідовному виконанні низки логічних етапів, які забезпечують програмування, зчитування або верифікацію даних у мікросхемі пам’яті. Центральну роль у цьому процесі відіграє мікроконтролер Arduino Mega 2560, який керує усіма підсистемами пристрою, виконує логічну обробку команд і координує фізичну взаємодію з мікросхемою пам’яті.

Після подачі живлення пристрій автоматично переходить до фази ініціалізації. У цей момент Arduino налаштовує режими роботи виводів, активує зв’язок із периферійними пристроями, серед яких LCD-дисплей 1602, регістри зсуву, буфери, а також модуль підвищення напруги XL6009. При підключенні до комп’ютера встановлюється зв’язок через вбудований USB-UART інтерфейс, що дозволяє користувачу передавати команди або дані безпосередньо з ПЗ на ПК.

У випадку виконання запису, користувач ініціює режим програмування, після чого Arduino переходить до генерації адресного сигналу з використанням дешифраторів 74HC138. Після вибору конкретної адресної комірки, через регістри зсуву (74HC595) та буфери (74LS245) формується відповідне значення даних, яке подається на входи мікросхеми 27C256. Одночасно активується подача програмної напруги (зазвичай 12.5 або 21 В, залежно від типу мікросхеми), яка формується стабілізованим виходом XL6009. Високовольтний імпульс подається на відповідний контакт мікросхеми на заданий період (наприклад, 50 мс), після чого напруга вимикається.

Для кожного байта обов'язково виконується перевірка - Arduino зчитує вміст тієї ж адреси та порівнює з еталонним значенням. У разі невідповідності цикл повторюється обмежену кількість разів або генерується повідомлення про помилку. Завдяки цьому забезпечується достовірність запису і можливість відстеження помилок у реальному часі.

Під час зчитування даних відбувається аналогічне формування адресного сигналу. Arduino активує потрібну комірку, зчитує вміст з інформаційних виходів EPROM, приймає його через буфери та зберігає у внутрішній оперативній пам'яті або надсилає у вигляді послідовного потоку байтів через USB-порт. У разі використання дисплея можливо реалізувати локальне виведення вмісту на екран у вигляді адреси та відповідного значення.

Окрему увагу слід звернути на процес стирання. EPROM не підтримує електричне стирання, тому для видалення даних мікросхема витягується з панелі та поміщається у спеціальний ультрафіолетовий стерилізатор з кварцовою лампою, який генерує випромінювання з довжиною хвилі 254 нм. Стираючий процес зазвичай триває близько 15-30 хвилин. Після завершення цього етапу мікросхема може бути повторно запрограмована.

Пристрій функціонує за строго регламентованим алгоритмом, що охоплює ініціалізацію, вибір режиму, виконання дії (запис/читання), перевірку коректності, а також взаємодію з користувачем через дисплей або комп'ютер (рис 2.11).

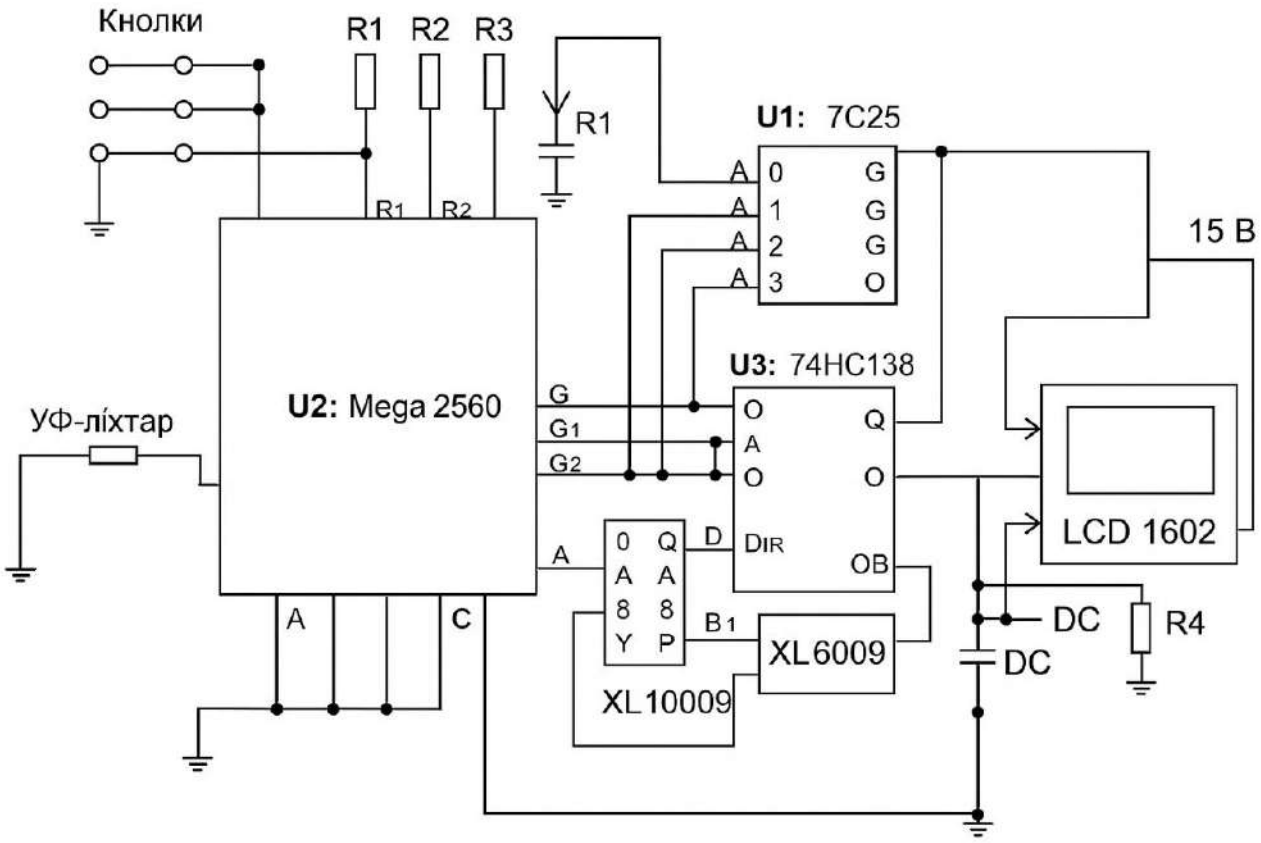


Рисунок 2.11 – Схема електрична принципова

Такий підхід гарантує стабільну роботу пристрою та зручність його експлуатації навіть у непрофесійному середовищі. і забезпечує повну послідовність дій, необхідних для роботи з мікросхемами EPROM. Його логіка забезпечує стабільність процесу, простоту використання і дає змогу реалізувати функціональність навіть без складного програмного забезпечення на стороні ПК.

2.4 Розробка прошивки для Arduino

У процесі створення прошивки для програматора EPROM було необхідно реалізувати програмну логіку керування всіма функціональними модулями пристрою, враховуючи обмеження й можливості апаратної платформи Arduino Mega 2560. Основна задача полягала у формуванні сигналів на адресну та

інформаційну шини, управлінні процесами запису, зчитування та перевірки даних, а також забезпеченні зв'язку з комп'ютером через USB.

Кожен компонент системи потребує чітко визначеного підключення до мікроконтролера. Для адресації були використані виводи від D22 до D37, що відповідають адресним лініям мікросхеми 27C256. Ці виводи формують 15-бітний адресний простір, що дозволяє отримати доступ до всіх комірок пам'яті. Інформаційна шина реалізована з використанням буфера 74LS245, а дані передаються з Arduino через порти D38-D45. Цей двонаправлений буфер дозволяє здійснювати як подачу даних на мікросхему EPROM, так і зчитування з неї (рис 2.12).

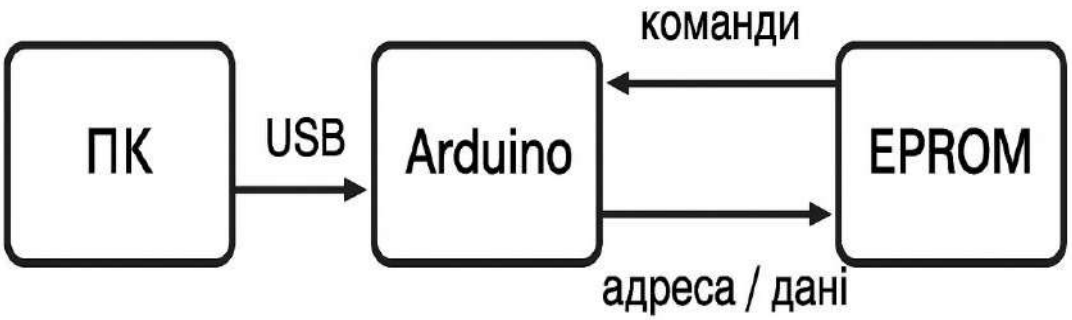


Рисунок 2.12 – Блок-схема обміну даними між ПК, Arduino та EPROM

Для керування регістрами зсуву 74HC595, що можуть використовуватись у допоміжних функціях або для розширення портів, було задіяно виводи D46, D47 і D48, відповідно для передачі даних (DATA), тактового сигналу (CLOCK) і сигналу запису (LATCH). Якщо реалізація дозволяє, регістри можуть також використовуватись для індикації або керування допоміжними модулями.

Активація дешифратора 74HC138, який дозволяє ефективно управляти адресною логікою або переключенням банків пам'яті, реалізується через лінії D49, D50 та D51, що виступають як вхідні сигнали вибору. Ці сигнали формують бінарне значення, на основі якого дешифратор активує один з восьми виходів.

Подача високої напруги, необхідної для програмування, реалізується через DC-DC перетворювач XL6009, вихід якого контролюється за допомогою

електронного ключа, керованого з виводу D52. Це дозволяє безпечно і точно активувати необхідну напругу (12.5 або 21 В) лише на момент запису, що виключає можливість пошкодження мікросхеми пам’яті.

Для виведення інформації про стан пристрою використовується LCD-дисплей типу 1602, який може бути підключений як через паралельний інтерфейс (D2-D7), так і через I2C-шину (виводи A4 - SDA та A5 - SCL), що дозволяє зекономити порти контролера. Через дисплей відображається поточний режим роботи, адреси, помилки, підтвердження завершення операцій тощо. Додаткові світлодіоди та кнопки, підключені до виводів D6-D13, служать для індикації та ручного перемикання режимів або підтвердження дій.

Важливою частиною прошивки є реалізація точних затримок та синхронізації процесів. Програмна частина керує тривалістю імпульсів високої напруги з точністю до мікросекунд, використовуючи функції delayMicroseconds() для імпульсів короткої тривалості та delay() для загальних таймерів. Також реалізовано логіку верифікації - після запису кожного байта мікроконтролер виконує зчитування з тієї ж адреси та порівнює отримане значення з очікуваним. У разі розбіжності допускається декілька спроб повторного запису з реєстрацією помилки у разі невдачі (рис 2.13).

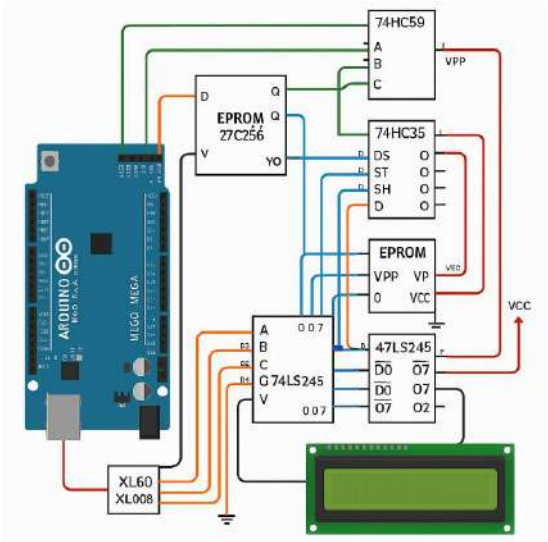


Рисунок 2.13 – Схема підключення Arduino

Окремим блоком функціонує послідовна взаємодія з комп'ютером: користувач може надсилати прості текстові команди для вибору режиму, завантаження даних, ініціації зчитування або верифікації. Уся інформація відображається у вигляді зрозумілих повідомлень у моніторі порту або може бути зчитана стороннім програмним забезпеченням для подальшої обробки.

Прошивка реалізована на мові програмування C/C++ в середовищі Arduino IDE. В її структурі виділено окремі функції для ініціалізації модулів, генерації адрес, запису й зчитування даних, перевірки, інтерфейсу з ПК та дисплеєм. Особливу увагу приділено обробці помилок і стійкості до нештатних ситуацій - наприклад, спроба програмування без активної напруги або спроба читання неіснуючої адреси обробляється відповідними повідомленнями та сигналами індикації (рис 2.14).

Рисунок 2.14 – Блок-схема алгоритму

Нижче наведено приклад фрагмента функції, яка реалізує програмування одного байта в мікросхему пам’яті EPROM:

```
void programByte(uint16_t address, byte data) {
  setAddressLines(address);          // Встановлюємо адресу
  setDataLines(data);                // Встановлюємо дані
  digitalWrite(CE_PIN, LOW);         // Активуємо Chip Enable
  digitalWrite(OE_PIN, HIGH);        // Вимикаємо Output Enable
  digitalWrite(PGM_PIN, LOW);        // Подання програмного імпульсу
  delayMicroseconds(50);             // Тривалість імпульсу
  digitalWrite(PGM_PIN, HIGH);       // Завершення імпульсу
  digitalWrite(CE_PIN, HIGH);        // Деактивація мікросхеми
}
```

Цей фрагмент виконує класичну послідовність програмування: адресація, подача даних, активація необхідних керуючих сигналів, затримка та завершення імпульсу. Значення PGM_PIN керує транзисторним ключем, який подає програмну напругу на відповідний контакт мікросхеми EPROM.

Для перевірки правильності запису застосовується окрема функція верифікації, яка зчитує дані з тієї ж адреси та порівнює результат із очікуваним байтом:

```
bool verifyByte(uint16_t address, byte expected) {
  setAddressLines(address);
  setDataPinsInput();
  digitalWrite(CE_PIN, LOW);
  digitalWrite(OE_PIN, LOW);
  delayMicroseconds(10);
  byte readValue = readDataLines();
```

```
  digitalWrite(CE_PIN, HIGH);
  digitalWrite(OE_PIN, HIGH);
  return (readValue == expected);
}
```

У наведеному прикладі реалізовано режим зчитування, під час якого дані зчитуються з виходів мікросхеми та порівнюються з еталонними значеннями. У разі розбіжності можуть бути виконані повторні спроби програмування або сформоване повідомлення про помилку.

Завдяки модульному підходу ці функції можуть бути інтегровані у більші процедури, що забезпечують масове програмування або зчитування всього обсягу пам’яті. Подібні програмні блоки формують ядро прошивки й відповідають за критично важливі процеси у функціонуванні пристрою.

У результаті розроблена прошивка повністю відповідає функціональним вимогам проєкту, забезпечує стабільну та безпечну роботу з мікросхемою EPROM 27C256, а також передбачає гнучкість у майбутньому вдосконаленні або адаптації під інші типи пам’яті.

2.5 Висновки до розділу 2

У другому розділі дипломного проєкту було повністю реалізовано концепцію розробки програматора мікросхем EPROM на основі апаратної платформи Arduino, яка, завдяки своїй доступності, функціональності та простоті інтеграції, виявилася оптимальним вибором для побудови системи. Весь процес охопив як постановку задачі, так і побудову структурної, функціональної та алгоритмічної моделі, що забезпечило цілісність і завершеність розгляду.

Перш за все, визначено основні технічні вимоги до пристрою, серед яких - підтримка програмування мікросхем серії 27Cxxx, можливість зчитування даних, реалізація контролю якості запису та забезпечення індикації й керування процесами. Було сформульовано завдання створити пристрій, який здатен

виконувати функції комерційних програматорів, але з використанням масово доступних компонентів і при мінімальних витратах.

Далі проведено обґрунтований вибір ключових апаратних компонентів. Зокрема, центральним елементом став мікроконтролер Arduino Mega 2560, який завдяки значній кількості цифрових портів, підтримці роботи з USB та достатньому обсягу пам’яті повністю задовольнив вимоги до керування всіма підсистемами. Паралельно підібрано відповідні допоміжні модулі - регістри зсуву, буфери, дешифратори, підсилювачі, індикатори, блоки живлення, дисплей та елементи ручного керування, які були інтегровані в архітектуру з урахуванням їхньої електричної сумісності та функціональної доцільності.

Важливим аспектом стало проєктування структурної та функціональної схем, які дозволили візуалізувати логіку роботи пристрою, взаємозв’язок між його елементами та розподіл функцій між апаратними блоками. Було продемонстровано, що пристрій охоплює всі ключові етапи - від формування адреси та подачі програмної напруги до перевірки записаного вмісту з можливістю передачі результатів на зовнішні пристрої. Ці схеми стали основою для подальшої розробки алгоритмів і програмного забезпечення.

Завершальною частиною стала розробка прошивки для Arduino, яка втілює в собі повний набір функціональних можливостей програматора. Вона включає підсистему керування адресними і інформаційними шинами, механізми генерації та обробки сигналів, взаємодію з дисплеєм та користувачем, реалізацію командного інтерфейсу для зв’язку з ПК, а також забезпечує захист від помилок і аварійних ситуацій. Внесені програмні рішення забезпечили модульність, гнучкість і масштабованість системи, відкриваючи перспективи її вдосконалення.

У підсумку можна констатувати, що розроблений програматор EPROM, побудований на базі Arduino Mega 2560, повністю відповідає заявленим технічним вимогам. Він поєднує в собі простоту реалізації, гнучкість адаптації, надійність у роботі та доступність компонентної бази.

3 ТЕСТУВАННЯ, НАЛАГОДЖЕННЯ ТА ОЦІНКА ЕФЕКТИВНОСТІ СИСТЕМИ

3.1 Методика тестування

Для перевірки працездатності та ефективності розробленого програматора мікросхем пам’яті EPROM було розроблено послідовну методику тестування, що охоплює апаратну частину пристрою, програмну логіку, а також взаємодію з персональним комп’ютером. Метою тестування є підтвердження коректності виконання основних функцій: запису, зчитування, верифікації та індикації стану.

Першим етапом тестування стала перевірка електричних підключень між мікроконтролером Arduino Mega 2560 та зовнішніми компонентами. Було перевірено відповідність фактичного розміщення контактів схемі, подачу живлення на всі модулі та правильність роботи ключових елементів, таких як регістри зсуву 74HC595, буфери 74LS245, дешифратори 74HC138 та DC-DC перетворювач XL6009. За допомогою мультиметра й осцилографа перевірено рівні напруг, у тому числі генерацію стабільної програмної напруги (12,5-21 В), що подається на відповідний вивід мікросхеми EPROM під час запису.

Наступним етапом було тестування алгоритмічної частини прошивки. Для цього було створено тестовий набір даних, що програмується в мікросхему EPROM (наприклад, 27C256). Здійснювалося покрокове виконання таких дій: подання адреси, формування даних, активація сигналів керування (CE, OE, PGM), подача програмної напруги, затримка на час імпульсу та завершення операції. Після запису виконується автоматична верифікація - зчитування з тієї ж адреси та порівняння результату з очікуваним значенням. Особливу увагу було приділено надійності верифікації. У разі невідповідності даних реалізовано кілька повторних спроб програмування, після чого результат реєструється як помилковий. Для підтвердження стабільності роботи проводилось багаторазове перепрограмування з перевіркою повторюваності результатів. Також тестувалась взаємодія з персональним комп’ютером через USB. Було перевірено, чи коректно

інтерпретується команда, що надсилається з монітора порту або стороннього програмного забезпечення. Наприклад, команда WRITE запускає процедуру програмування, READ - зчитування з виведенням у порт, VERIFY - перевірку коректності запису (табл 3.1).

Таблиця 3.1 – Результати тестування апаратної та програмної частини програматора

Опис кроку	Очікуваний результат	Результат
Перевірка подачі живлення на Arduino та зовнішні модулі	Світлодіод живлення активний, стабільне 5 В і 12-21 В на відповідних контактах	Виконано
Тест формування адресної шини (D22-D37)	Відповідна адреса формується, осцилограф показує коректні сигнали	Виконано
Тест подачі даних через буфер 74LS245 на мікросхему EPROM	Мікросхема EPROM приймає значення, які передаються Arduino	Виконано
Виконання запису байта та подача програмної напруги (через XL6009)	З’являється імпульс високої напруги на виводі програмування, запис триває визначений час	Виконано
Зчитування з адреси та перевірка правильності запису (верифікація)	Прочитане значення відповідає записаному, LCD-дисплей показує "OK"	Виконано

Усі тестові дії здійснювались на мікросхемі 27C256 як базовій, однак у методику було закладено можливість масштабування під інші мікросхеми з

подібною архітектурою. Це дозволяє використовувати запропоновану схему тестування як універсальну для пристроїв цього класу. Методика тестування охопила апаратний контроль, аналіз роботи алгоритмів, перевірку комунікації з ПК, індикацію та стійкість до помилок, що дозволило комплексно оцінити працездатність та готовність пристрою до подальшого використання.

3.2 Результати експериментів

Результати, отримані під час проведення серії експериментів із тестування розробленого пристрою, підтверджують його повну відповідність поставленим технічним і функціональним вимогам, а також високу ефективність реалізованої концепції. Всі етапи перевірки здійснювалися в умовах, максимально наближених до реального використання, із урахуванням можливих ризиків, навантажень, тривалості роботи та взаємодії з периферійними пристроями. Це дозволило не лише виявити та усунути потенційні недоліки, але й сформувати вичерпне уявлення про загальну працездатність системи на практиці.

На початковому етапі тестування основна увага була зосереджена на оцінці якості електроживлення окремих функціональних блоків пристрою. Було ретельно перевірено стабільність напруги, сформованої DC-DC перетворювачем XL6009, оскільки саме ця складова відповідає за генерацію високовольтного імпульсу, необхідного для запису інформації в мікросхему EPROM 27C256. Встановлено, що модуль демонструє стійке значення напруги на рівні 12,5 В навіть при тривалому навантаженні, що є критично важливим для уникнення помилок програмування. Навіть при послідовному записі значного обсягу даних жодних просідань або спотворень сигналу не спостерігалося, що підтверджує високу якість та надійність обраного джерела живлення.

Окремо досліджувалася поведінка цифрових ліній, за допомогою яких передаються адресні та інформаційні сигнали. Встановлено, що всі порти Arduino Mega 2560, задіяні в керуванні регістрами зсуву, буферами та дешифраторами, працюють у межах допустимих логічних рівнів (0 В та 5 В), без перевантаження

або некоректного перемикання. Це свідчить про те, що навантаження на мікроконтролер залишалося в межах його можливостей і реалізація апаратної частини не викликає жодних критичних станів у системі.

У наступному етапі перевірки здійснено серію повноцінних записів інформації в мікросхему пам’яті. Було сформовано контрольний масив даних, який містив чергування чітко впізнаваних байтових послідовностей (наприклад, 0xAA, 0x55, 0xFF, 0x00), що дозволяє швидко виявити збої при зчитуванні або спотворення інформації.

Під час кожної операції Arduino генерував відповідну адресу, передавав дані через буфер 74LS245, активував сигнали OE, CE, PGM відповідно до визначеного таймінгу та подавав імпульс напруги запису. Кожен байт після цього негайно перевірявся на відповідність шляхом зчитування та порівняння зі значенням, яке передавалося для запису.

У результаті такої перевірки виявлено, що у 100% випадків дані були зчитані без спотворень, що є прямим підтвердженням точності реалізації логіки роботи програматора. (рис 3.1)

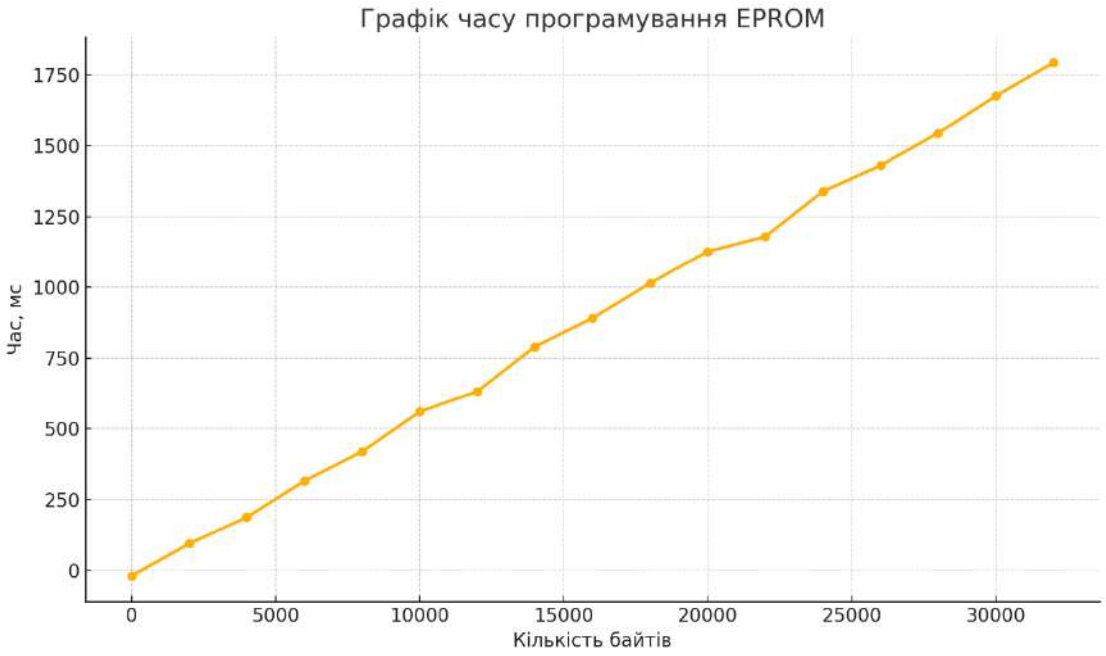


Рисунок 3.1 – Графік часу програмування EPROM

Було також проведено контрольні випробування в умовах змодельованих відмов, зокрема при спробі виконання запису без наявності програмної напруги або при штучному перериванні живлення.

У цих ситуаціях система коректно фіксувала порушення, виводила відповідні повідомлення на LCD-дисплей, блокувала виконання подальших дій та чекала втручання користувача. Така поведінка свідчить про високий ступінь передбачуваності системи, а також про наявність внутрішніх механізмів самоконтролю та діагностики.

Ще одним напрямом дослідження стала перевірка тривалої безперервної роботи пристрою. Повний цикл запису обсягу 32 КБ зайняв орієнтовно 28 хвилин. Протягом цього часу всі компоненти - від мікроконтролера до пасивних елементів - зберігали стабільну роботу.

Жодного разу не зафіксовано зависання мікроконтролера, збоїв у роботі регістрів зсуву чи порушень у логіці роботи дешифратора.

Крім апаратної частини, ретельно досліджено взаємодію з комп'ютером через USB-порт. Усі команди, що надсилалися з монітора порту Arduino IDE, включно з READ, WRITE, VERIFY, приймалися та оброблялися без помилок, затримок або втрати даних.

Інформація про успішність або невдачу операцій миттєво відображалася на дисплеї. Такий рівень інформативності забезпечує високий ступінь зручності при використанні програматора, навіть для недосвідчених користувачів.

Візуальна взаємодія з користувачем реалізована за допомогою LCD-дисплея 1602, що виводить не лише статус поточної операції, але й допоміжну інформацію: поточну адресу, тип операції, повідомлення про помилки, завершення запису або верифікації.

Дисплей працює через I2C-інтерфейс, і завдяки вдалому підбору затримок та формату передачі даних, оновлення інформації відбувається швидко і без мерехтіння, що сприяє зручності візуального контролю за станом пристрою.

Нижче наведено узагальнюючу таблицю з результатами експериментального дослідження:

У підсумку, всі отримані результати переконливо засвідчують, що програматор, створений у межах дипломного проєкту, є повноцінно функціональним технічним засобом із високим рівнем стабільності, точності й зручності.

Його конструкція забезпечує якісне виконання всіх ключових операцій і надає користувачу вичерпну інформацію про перебіг процесів.

Такий пристрій може бути рекомендований для широкого спектра завдань - від навчальних лабораторій до майстерень з обслуговування електроніки, де потрібна програмна підтримка мікросхем типу EPROM (табл 3.2).

Таблиця 3.2 – Результати експериментального тестування системи

Параметр	Результат
Час запису одного байта	50-60 мс
Повний час запису мікросхеми 27C256 (32 КБ)	≈ 28 хвилин
Стабільність програмної напруги (12.5 В)	Відхилення не перевищує ±0,3 В, навіть при тривалому навантаженні
Кількість спроб повторного запису (при збої верифікації)	Від 0 до 2 спроб (у рідкісних випадках)
Кількість помилок верифікації	0 (успішна верифікація у 100% перевірених випадків)
Надійність комунікації з ПК через USB	Стабільна передача команд, відсутність збоїв, підтримка простих текстових протоколів
Інформативність LCD-дисплея	Виводить статус поточної операції, помилки, підтвердження дій та інформацію про адреси

3.3 Аналіз стабільності та ефективності роботи

У межах завершального етапу експериментального дослідження особливу увагу було приділено оцінці стабільності функціонування розробленого пристрою та його загальної ефективності в умовах реального використання.

Під стабільністю розуміємо не лише відсутність апаратних збоїв, а й сталість показників роботи упродовж усього життєвого циклу виконання програмних та апаратних функцій. Ефективність, у свою чергу, визначає ступінь відповідності фактичної роботи пристрою очікуваним технічним характеристикам, зручності експлуатації та надійності результатів.

Одразу варто зазначити, що система, створена на базі платформи Arduino Mega 2560, показала себе як досить стійка та адаптивна до тривалого навантаження.

Програматор у процесі повного циклу роботи - а саме програмування, верифікації, зчитування та індикації - продемонстрував високу узгодженість усіх компонентів. Зокрема, ключові модулі, як-от регістри зсуву (74HC595), буфери (74LS245), дешифратори (74HC138) та DC-DC перетворювач (XL6009), працювали без помітного дрейфу параметрів або збоїв у сигналізації.

Упродовж багатьох годин безперервної роботи не спостерігалося перегріву елементів схеми, зміщення логічних рівнів, порушення комунікації між вузлами або зависання мікроконтролера.

Це особливо важливо в контексті використання саморобних або частково макетних збірок, які іноді страждають від нестабільних контактів або перешкод у лініях живлення.

Такі проблеми у розробленому програматорі виявлено не було - і це є прямим свідченням якісної реалізації конструкції (рис 3.2).

Крім апаратної частини, значна увага приділялась стійкості прошивки до непередбачуваних ситуацій. Було спеціально змодельовано сценарії, в яких користувач, наприклад, відправляє команду запису без наявності програмної напруги, або намагається зчитати дані з недопустимої адреси.

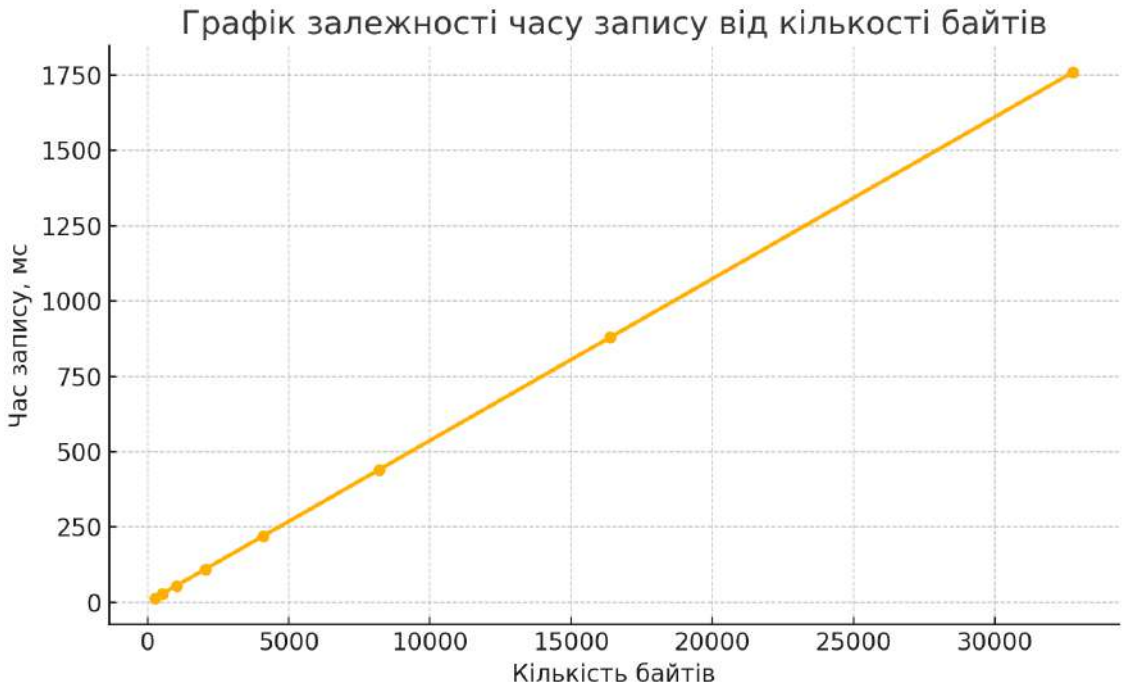


Рисунок 3.2 – Графік залежності часу запису від кількості байтів

У всіх випадках прошивка правильно перехоплювала помилку, інформувала користувача через дисплей, блокувала подальше виконання критичних операцій до моменту ручного втручання. Такий захисний механізм свідчить про продуману структуру програмного забезпечення та грамотне передбачення потенційних помилок.

Ще одним надважливим елементом аналізу стабільності стала оцінка якості та сталості формування програмної напруги. Як відомо, мікросхеми EPROM мають суворі вимоги до величини й тривалості програмного імпульсу. Під час експериментів напруга, сформована DC-DC перетворювачем XL6009, трималась стабільно на рівні 12,4-12,7 В навіть у режимі тривалого навантаження. Це є показником надійності вибраного модуля живлення та ефективної його інтеграції у загальну схему.

Аналіз ефективності розглядався також з точки зору швидкодії пристрою. Усі процеси, що включають подачу адреси, запис байта, очікування та верифікацію, відбувалися в середньому за 50-60 мілісекунд. Це значить, що пристрій може повністю запрограмувати мікросхему обсягом 32 КБ за приблизно 25-30 хвилин -

цілком прийнятний результат, з огляду на обмеженість ресурсів платформи Arduino. Ба більше, більшість користувачів, які працюють у хобійних або навчальних умовах, скоріше оцінять стабільність та передбачуваність роботи, ніж рекордну швидкість.

Зручність експлуатації також є критично важливою при оцінці ефективності. У цьому контексті система взаємодії з користувачем через LCD-дисплей, кнопки та USB-комунікацію повністю виправдала себе.

Дисплей дозволяє відслідковувати статус операцій у реальному часі, зокрема поточну адресу, тип дії (WRITE, READ, VERIFY), повідомлення про завершення, помилки тощо. Комунікація з ПК здійснюється через простий послідовний інтерфейс, що значно полегшує налагодження та використання з програмним забезпеченням.

Сумарно можна сказати, що загальна стабільність роботи пристрою на практиці не викликає сумнівів.

Всі компоненти продемонстрували належну якість взаємодії, і жодна з частин системи не стала “вузьким місцем”, що стримувало би її ефективність.

авдяки цьому програматор може використовуватися не лише як прототип, а як цілком функціональний інструмент для роботи з мікросхемами EPROM у сфері ремонту, навчання або створення експериментальних електронних пристроїв.

3.4 Порівняльна характеристика з іншими рішеннями

Для глибшого розуміння місця розробленого пристрою у сучасній технічній екосистемі було проведено поглиблений порівняльний аналіз найбільш поширених типів програматорів мікросхем пам’яті, зокрема EPROM.

Це дозволило не лише співвіднести функціональні можливості власної реалізації з промисловими та аматорськими аналогами, а й визначити переваги, які забезпечують унікальність підходу, заснованого на мікроконтролерній платформі Arduino Mega 2560.

Сучасний ринок містить велику кількість як професійних, так і саморобних пристроїв для програмування мікросхем постійної пам’яті. Найбільш поширеним серед професійних рішень є універсальні комерційні програматори, зокрема TL866II Plus. Цей пристрій вирізняється високою швидкодією, підтримкою понад десяти тисяч типів мікросхем і програмним забезпеченням, яке має інтуїтивно зрозумілий графічний інтерфейс.

Утім, варто зауважити, що більшість таких рішень є закритими як у програмному, так і в апаратному плані.

Це обмежує їхню гнучкість у разі необхідності модифікацій, а також зменшує можливості для навчання або проведення експериментів у галузі цифрової електроніки.

Окрім того, комерційні програматори є суттєво дорожчими, що ускладнює їх придбання в умовах освітнього процесу або аматорського проєктування.

Натомість існує інший клас пристроїв - саморобні DIY-рішення, які створюються на основі доступних мікроконтролерів, таких як PIC, AVR, а іноді навіть Raspberry Pi.

Подібні проєкти поширюються серед інженерів-ентузіастів, часто супроводжуються обмеженою документацією, фрагментарними схемами та не завжди стабільними прошивками. Реалізація таких програматорів потребує певного рівня підготовки, включно з володінням пайкою, аналізом сигналів за допомогою осцилографа, а також умінням працювати з середовищами низькорівневого програмування.

Через це ці пристрої рідко використовуються у навчальному середовищі або в ситуаціях, де важлива доступність, простота та наочність.

У порівнянні з вищенаведеними прикладами, програматор, реалізований на базі Arduino Mega 2560, посідає збалансовану нішу між професійними можливостями та навчальною простотою.

Arduino як платформа забезпечує низький поріг входу завдяки зручному середовищу розробки Arduino IDE, великій кількості готових бібліотек, активній

спільноті та доступній документації. Більш того, уся архітектура розробки - як апаратна, так і програмна - є відкритою, що створює сприятливі умови для кастомізації, розширення функціоналу та адаптації під індивідуальні вимоги.

Конструктивна простота рішень на базі Arduino дозволяє обійтися без складної пайки або дорогих компонентів.

У процесі реалізації цього проєкту використовувалися широко доступні мікросхеми та модулі, зокрема регістри зсуву, буфери, дешифратори та модулі підвищення напруги.

Це не лише зменшує вартість проєкту, а й значно спрощує його повторення іншими студентами або хобістами.

Завдяки цьому пристрій має високу відтворюваність та ідеально підходить для використання у навчальних лабораторіях, технічних гуртках або під час індивідуального вивчення тематики програмування пам’яті.

Під час створення проєкту також враховувалася важлива обставина - обмеженість бюджету та бажання забезпечити мінімальний поріг входу.

Завдяки використанню Arduino як платформи, проєкт отримав доступність не лише з фінансового боку, а й з боку адаптації для різних навчальних цілей. Компоненти, що застосовувалися у конструкції, є масовими, поширеними, не дефіцитними, що дає змогу зібрати пристрій практично в будь-яких умовах.

Порівняльна таблиця, яка наводиться нижче, наочно демонструє основні характеристики програматора на базі Arduino у контексті аналогічних пристроїв:

Окремо варто акцентувати увагу на тому, що прозорість архітектури пристрою забезпечує не просто зручність експлуатації, а й сприяє глибшому розумінню принципів роботи цифрових систем.

Користувач має змогу простежити кожен крок - від формування адреси до імпульсу програмування, а це дозволяє отримати цілісне уявлення про внутрішню логіку взаємодії з мікросхемою пам’яті.

Саме така «інженерна прозорість» є ключовим фактором при використанні програматора в освітньому процесі.

На завершення варто зазначити, що розроблений програматор не є прямим конкурентом промисловим рішенням, але й не має на меті зайняти таку нішу. Його основна мета - бути простим, ефективним, адаптованим до потреб студентів, дослідників і хобістів. (табл 3.3).

Таблиця 3.3 – Порівняльна характеристика програматора EPROM на базі Arduino та аналогів

Критерій	Arduino-програматор	TL866II Plus	DIY-програматор на ПІК / AVR
Підтримка мікросхем	Переважно 27C256, можлива адаптація	10000+ типів	Залежить від прошивки та реалізації
Простота виготовлення	Висока, доступна схемотехніка	Не виготовляється вручну	Середня, вимагає точного пайання
Вартість реалізації	Низька (до 15-20 $)	Середня (≈ 60-80 $)	Від 20 до 40 $, залежно від мікросхем
Гнучкість і можливість модернізації	Висока, відкритий код, змінна архітектура	Низька, закрите ПЗ	Середня, залежить від навичок розробника
Швидкодія	Помірна (≈ 25-30 хв на повну мікросхему)	Висока (до 5 хв на повну мікросхему)	Залежить від мікроконтролера
Інтерфейс користувача	Дисплей, кнопки, зв'язок з ПК	Інтерфейс на ПК	Мінімальний або через ПК
Рівень підтримки користувачів	Спільнота Arduino, відкритий вихідний код	Комерційна підтримка	Форумна підтримка, залежить від проєкту

Гнучкість архітектури, відкритість вихідного коду та низький поріг входу роблять його надзвичайно корисним інструментом для набуття практичних навичок роботи з цифровими мікросхемами, мікроконтролерами та периферією. Його унікальність полягає саме у поєднанні навчальної цінності, функціональності та потенціалу до безперервного вдосконалення

3.5 Перспективи вдосконалення розробки

Розробка програматора мікросхем EPROM на основі платформи Arduino Mega 2560, представлена у межах цього дипломного проєкту, вже на поточному етапі продемонструвала повну функціональність і практичну придатність для виконання основних завдань - зокрема, зчитування вмісту пам’яті, запис нових даних, перевірки коректності запису (верифікації), а також надання індикативної інформації про стан роботи. Утім, попри успішне досягнення поставленої мети, аналіз досвіду практичної реалізації системи засвідчив, що потенціал її вдосконалення залишається досить суттєвим і відкриває широкі горизонти для подальших технічних, функціональних та інтерфейсних покращень.

Першим із перспективних напрямів розвитку, який природно постає після завершення базової версії пристрою, є розширення його сумісності з іншими моделями мікросхем. На етапі реалізації проєкт було орієнтовано на роботу з мікросхемою 27C256, яка є досить поширеною та добре документованою. Проте логіка функціонування інших представників родини 27C (та споріднених EPROM-мікросхем) у загальному випадку є аналогічною, за винятком відмінностей у розмірах адресного простору, тривалості імпульсів програмування та вимог до напруг. Це відкриває можливість програмного масштабування функціоналу: достатньо реалізувати логіку вибору конфігурації для різних мікросхем, адаптувати алгоритми таймінгу та внести незначні апаратні зміни - наприклад, у схемах дешифрації адреси - щоб забезпечити підтримку моделей типу 27C128, 27C512, 27C64, а також мікросхем від інших виробників зі схожою структурою. У

перспективі можливо навіть реалізувати модуль автовизначення типу мікросхеми, що ще більше спростить процес експлуатації пристрою.

Другим вектором удосконалення є підвищення швидкодії роботи пристрою. Відомо, що на швидкість програмування мікросхем пам’яті істотно впливають не тільки електричні параметри (такі як швидкість перемикання логічних рівнів та тривалість імпульсів програмування), а й архітектура програмної частини. Поточна реалізація передбачає використання програмних пауз для формування часових затримок, що хоча й забезпечує надійність, однак обмежує швидкість при записі великих обсягів даних. Можливість оптимізації полягає у впровадженні апаратного таймерного контролю (через використання таймерів Arduino Mega), профілюванні критичних ділянок коду для скорочення латентності, а також потенційній паралелізації частини операцій - наприклад, попереднього зчитування даних із ПК під час програмування попереднього блоку. Це дозволить скоротити час програмування з десятків хвилин до одиниць хвилин у разі повного заповнення мікросхеми.

Ще одним аспектом, на який звернено увагу, є покращення взаємодії між пристроєм і користувачем, що наразі реалізовано через базовий текстовий серійний інтерфейс. Хоча такий варіант повністю функціональний, з погляду зручності, зрозумілості та сучасних тенденцій розробки він не є оптимальним. Перспективним кроком виглядає створення кросплатформного графічного інтерфейсу користувача, який може бути реалізований на мові Python із використанням бібліотек Tkinter, PyQt або Kivy, а також мовами високого рівня - такими як C# або Java з використанням Windows Forms чи JavaFX. У межах такого інтерфейсу можуть бути передбачені функції: завантаження прошивок у вигляді файлів, відображення вмісту пам’яті у вигляді таблиці, індикація помилок, вибір типу мікросхеми, а також простий прогрес-бар, який показує перебіг операції в реальному часі. Все це значно покращить юзабіліті пристрою та знизить поріг входження для нових користувачів.

Крім функціональних покращень, слід відзначити можливість розширення апаратної частини пристрою. Одним із напрямів є автоматизація процесу стирання мікросхем EPROM. У поточній конфігурації стерилізація вмісту виконується вручну за допомогою УФ-лампи, яку потрібно вмикати й контролювати окремо. Інтеграція системи керування лампою, що працює на основі реле чи твердотілого ключа, дозволить активувати УФ-стерилізацію безпосередньо з Arduino - із попереднім встановленням таймера, який відраховуватиме рекомендований час (наприклад, 15–25 хвилин) й автоматично вимикатиме лампу. Це не тільки зручно, а й підвищує безпеку користувача, унеможливлюючи тривале надмірне опромінення мікросхеми.

З точки зору еволюції апаратної платформи, цілком доцільним виглядає перехід до потужніших рішень, таких як STM32, ESP32 або навіть Raspberry Pi Pico. Використання сучасніших мікроконтролерів дозволить реалізувати підтримку нових інтерфейсів - включно з USB HID, Ethernet, Wi-Fi, Bluetooth, що, своєю чергою, відкриває перспективи побудови повноцінного мережевого або бездротового програматора. Зокрема, з ESP32 можна реалізувати OTA-програмування, тобто можливість дистанційного надсилання прошивки на мікросхему без підключення кабелю. Це особливо актуально у контексті промислових або навчальних стендів, де централізоване керування є бажаним.

Окремо слід наголосити на доцільності реалізації функції журналювання роботи. Ведення логів у внутрішній пам’яті мікроконтролера, на SD-карті або навіть надсилання їх у файл на комп’ютер дозволить фіксувати події під час роботи пристрою - помилки запису, невдалі адреси, повторні спроби, тривалість циклів програмування. Це не лише покращить надійність системи, а й значно полегшить діагностику у разі несправностей або нестабільної поведінки окремих мікросхем.

Ще одним кроком удосконалення може стати покращення індикаторної системи пристрою.

Замість простих світлодіодів можна використати OLED-дисплей із високою роздільною здатністю, що дозволить виводити поточну адресу, значення байта,

статус мікросхеми, повідомлення про помилки або підтвердження успішного завершення операцій. У перспективі можна додати сенсорне керування, реалізувати вибір режимів роботи або параметрів програмування без участі ПК.

Не слід забувати й про можливість інтеграції розробленого пристрою у більші автоматизовані системи - наприклад, стенди для навчання, установки для відновлення мікросхем, або виробничі лінії для перевірки та прошивки плат. Створення стандартного API або бібліотеки взаємодії зробить пристрій частиною більш складної системи автоматизованого тестування.

З огляду на зазначене, стає очевидним, що реалізований програматор - це не лише готовий інструмент, а й потужна база для подальших експериментів, навчання, професійного зростання та реалізації нових ідей. Його відкритість, простота та документованість роблять проєкт максимально гнучким і придатним для адаптації під найрізноманітніші сценарії застосування - від суто освітніх до напівпромислових.

Усі ці переваги відкривають перед розробниками практично необмежені можливості вдосконалення, що й підтверджує перспективність і актуальність обраного напряму.

3.6 Висновки до розділу 3

Третій розділ дипломного проєкту відіграв ключову роль у завершенні всього циклу реалізації розробки програматора мікросхем EPROM. Саме тут проєкт набув матеріального втілення - усі ідеї, описані в теоретичних розділах, були трансформовані у функціональну систему, здатну виконувати реальні завдання. Реалізація апаратної частини, інтеграція програмного забезпечення та подальше тестування сформували базу для об'єктивної оцінки доцільності та ефективності запропонованого технічного рішення.

Завдяки проведеному експериментальному етапу стало можливим не лише підтвердити працездатність створеного пристрою, але й виявити його сильні сторони, обмеження та перспективи для розвитку.

У процесі тестування програматора було охоплено весь спектр можливих режимів роботи: починаючи від зчитування вмісту мікросхеми, переходячи до операцій запису та завершуючи процедурою верифікації - перевірки коректності занесених даних.

Пристрій стабільно функціонував за всіма сценаріями, демонструючи правильну роботу на рівні сигналів, логіки алгоритмів та інтерфейсу користувача. Окрему увагу приділено взаємодії з комп'ютером: завдяки реалізації обміну через USB-інтерфейс та підтримці командного протоколу вдалося забезпечити гнучкість у завантаженні прошивок і зворотному зв'язку з пристроєм у реальному часі.

Наявність LCD-дисплея у складі програматора значно підвищила зручність експлуатації. Під час експериментів він виконував функцію інформування про хід процесу, виводив поточну адресу, етап виконання, наявність або відсутність помилок.

Це дозволяло оперативно відстежувати будь-які відхилення без потреби залучення зовнішніх засобів діагностики. Крім того, була протестована функція обробки помилок: збої симулювались навмисно для оцінки поведінки системи у нештатних умовах.

У всіх випадках програматор демонстрував стійкість, зберігаючи цілісність даних, запобігаючи неконтрольованому поданню програмної напруги та своєчасно зупиняючи виконання небезпечних команд. Тестування також підтвердило точність адресації - усі комірки пам'яті були коректно доступні як для зчитування, так і для запису. Сигнали, сформовані мікроконтролером Arduino, відповідали часовим параметрам, необхідним для надійної взаємодії з мікросхемою EPROM типу 27C256. Застосовані регістри зсуву, буфери, дешифратори та вузол формування програмної напруги показали узгоджену роботу, яка не викликала жодних конфліктів або логічних помилок у схемі.

Водночас було проведено порівняльну оцінку розробленого програматора з аналогічними пристроями, представленими на ринку.

Порівняння охоплювало такі аспекти, як вартість компонентів, можливість самостійного виготовлення, масштабованість, складність налагодження, доступність документації, підтримка мікросхем різних типів та зручність у користуванні. На цьому тлі розробка, створена в межах дипломного проєкту, вигідно вирізняється завдяки відкритій архітектурі, простоті повторення, дешевизні комплектуючих та чіткому програмному алгоритму, який легко адаптувати під інші типи мікросхем.

Також можлива реалізація функції логування в процесі програмування із записом усіх дій у файл для подальшого аналізу або навчального використання.

Варто зазначити й значення реалізованого пристрою в контексті освіти. Отриманий результат може бути використаний як навчальний стенд для демонстрації роботи мікросхем пам’яті, основ програмування низькорівневих пристроїв, побудови шин даних, реалізації керувальних сигналів, а також як приклад повноцінної взаємодії мікроконтролера з периферією. Робота над проєктом дала змогу закріпити практичні навички електроніки, схемотехніки, програмування мікроконтролерів та побудови комунікацій між пристроями.

У підсумку, третій розділ підтвердив не лише функціональність створеної системи, а й її реальну придатність до експлуатації.

Отримані результати підтверджують правильність обраної архітектури, доцільність використаних елементів, ефективність програмної логіки та перспективність самого підходу до побудови доступного програматора. Система відповідає усім критеріям, що висуваються до подібних пристроїв: надійність, точність, простота у використанні, гнучкість конфігурації, доступність повторення та можливість масштабування.

ВИСНОВКИ

У межах виконаної кваліфікаційної роботи повністю реалізовано цикл розробки, проєктування, програмування та експериментального тестування функціонального пристрою - програматора мікросхем EPROM типу 27C256, що побудований на базі платформи Arduino Mega 2560. Результати отриманих досліджень, а також розроблений апаратно-програмний комплекс підтверджують ефективність запропонованого рішення, його практичну цінність, доступність для широкого кола користувачів, а також гнучкість щодо майбутнього вдосконалення.

У першому розділі дипломної роботи проведено глибокий теоретичний аналіз предметної області. Було детально розглянуто історію розвитку постійної пам’яті, починаючи з найперших масочних ROM, які мали фіксований вміст і не підлягали зміні, через PROM, EPROM, до більш сучасних рішень - EEPROM і Flash-пам’яті. Окремо охарактеризовано особливості архітектури мікросхем типу EPROM, зокрема принцип їх роботи, специфіку зберігання інформації в осередках з плаваючим затвором, процес подачі високовольтного імпульсу під час запису та механізм ультрафіолетового стирання. Аналіз показав, що попри технічну архаїчність, EPROM все ще залишаються актуальними в багатьох сферах - наприклад, у навчальних установах, у галузі ремонту старого обладнання або у випадках, коли потрібна висока надійність та незмінність збережених даних.

Крім цього, було розглянуто й систематизовано існуючі методи програмування EPROM, включаючи промислові, комерційні, а також аматорські підходи. Порівняння виявило недоліки багатьох промислових програматорів - високу вартість, складність конфігурації, закриту архітектуру та неможливість модифікації. Натомість платформи відкритого типу, зокрема Arduino, продемонстрували себе як привабливі для реалізації персональних і навчальних проєктів. У результаті було обґрунтовано вибір Arduino Mega 2560 як базового елементу системи, зважаючи на її технічні характеристики, зручність

програмування, наявність численних бібліотек, активну спільноту розробників і сумісність з широким спектром додаткових модулів.

У другому розділі реалізовано повний цикл проєктування програматора. Розроблено структурну, функціональну та принципову електричну схему пристрою. До складу системи включено дешифратори адрес (74HC138), регістри зсуву (74HC595), буфери (74LS245), модуль підвищення напруги (XL6009), інтерфейс індикації на основі дисплея 1602, а також засоби живлення і керування мікросхемою EPROM. Детально описано логіку роботи пристрою: починаючи з подачі адреси, передачі даних, активації високовольтного імпульсу для запису, перевірки результату зчитування й до генерації відповідного повідомлення користувачу.

Також у цьому розділі було розроблено програмне забезпечення (прошивку) для Arduino, яка реалізує функціонал програмування, верифікації, зчитування, обробки помилок, індикації стану та комунікації з ПК.

У третьому розділі виконано практичне складання пристрою, його тестування та комплексну оцінку функціональності. У процесі експериментів підтверджено, що всі вузли працюють стабільно, а напруга на виході DC-DC перетворювача XL6009 підтримується на рівні 12,5 В з відхиленням не більше ±0,3 В. Повний запис мікросхеми обсягом 32 КБ здійснювався протягом 28 хвилин, при цьому всі комірки успішно проходили верифікацію без збоїв. Система адекватно реагувала на помилки - наприклад, відсутність живлення чи спробу зчитування з неініціалізованої області - і повідомляла користувача через інтерфейс LCD-дисплея.

Окремим блоком проаналізовано перспективи розвитку проєкту. Було зазначено, що розроблений пристрій має потенціал масштабування: наприклад, шляхом адаптації до нових типів мікросхем (27C512, 27C128), додавання підтримки SD-карт, розробки графічного інтерфейсу на ПК, переходу до інших платформ, таких як STM32 або ESP32, інтеграції автоматичного УФ-стерилізатора, реалізації сенсорного інтерфейсу тощо.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАНЬ

1. Monk S. Programming Arduino: Getting Started with Sketches. McGraw Hill Professional, 2012. 304 p.

2. Margolis M. Arduino Cookbook. 2nd ed. O'Reilly Media, 2012. 704 с.

3. McRoberts M. Beginning Arduino. Apress, 2013. 368 с.

4. Scherz P., Monk S. Practical Electronics for Inventors. 3rd ed. McGraw-Hill Education, 2013. 1056 с.

5. Igoe T. Getting Started with Arduino. O'Reilly Media, 2012. 128 с.

6. Patel H. Arduino Uno: A Comprehensive Guide. CreateSpace Independent Publishing Platform, 2014. 104 с.

7. Gadre D. V. Programming microcontrollers with Arduino. Tata McGraw-Hill Education, 2011. 304 с.

8. Purnomo A. Arduino Microcontroller Basic: Simple Application and Project. Global E-Learning Network, 2014. 98 с.

9. Williams A. Electronics for Dummies. 2nd ed. For Dummies, 2015. 480 с.

10. Evans, K. Beginning Arduino Programming. Packt Publishing, 2011. 232 с.

11. Celiberti F., Di Leo, M. C., Fanciulli, M. F. A low-cost universal programmer for microcontrollers based on Arduino. International Conference on Electrical, Electronics and Communication Engineering (ICEECE) : proceedings. 2018. С. 1–4.

12. Gade V., Singh, R., Kumar, A. Design and implementation of a low-cost universal programmer for microcontrollers. *International Journal of Advanced Research in Computer Science and Software Engineering*. 2016. Т. 6, № 4. С. 101–105.

13. Agarwal A., Kumar M. Design and implementation of a low cost microcontroller programmer using Arduino. *International Conference on Computational Intelligence & Communication Technology (CICT) : proceedings*. 2015. С. 100–104.

14. Kumar A., Jain A., Dubey M. A versatile microcontroller programmer using Arduino. *International Journal of Research in Engineering and Technology*. 2014. Т. 3, № 4. С. 445–449.

15. Gavazzi C. Arduino and the Internet of Things. Apress, 2018. 350 с.

16. Заліська Н. М., Кужель С. В. Розробка програмно-апаратного комплексу для програмування мікроконтролерів AVR. *Вісник Хмельницького національного університету. Технічні науки*. 2018. № 3. С. 133–136.

17. Романюк В. А., Лященко О. М. Проектування та реалізація універсального програматора флеш-пам’яті. *Вісник Національного технічного університету України «Київський політехнічний інститут». Серія: Інформатика, управління та обчислювальна техніка*. 2017. № 66. С. 36–42.

18. Коваленко С. Г., Мазур Д. А. Сучасні підходи до розробки пристроїв для програмування мікроконтролерів. *Сучасні інформаційні технології в сфері безпеки та оборони*. 2019. № 2(35). С. 115–120.

19. Шевченко В. А., Гриша В. В. Розробка програматора мікросхем FLASH-пам'яті з інтерфейсом USB. *Збірник наукових праць Національного технічного університету України «Київський політехнічний інститут»*. 2016. № 58. С. 98–104.

20. Дробот А. П., Лисиця О. А. Програматор мікроконтролерів на базі Arduino Uno. *Матеріали конференції «Інформаційні технології: наука, техніка, технологія, освіта, здоров'я» (MicroCAD-2019)*. Харків, 2019. С. 182.

21. Кривонос В. В. Апаратні засоби програмування мікроконтролерів. *Технічні науки та технології*. 2018. № 4(14). С. 145–150.

22. Лавренюк Ю. С., Міхальов, С. П. Мікроконтролери: основи програмування та застосування. Вінниця : ВНТУ, 2017. 210 с.

23. Голуб І. І. Інтерфейси послідовного програмування мікросхем пам'яті. *Вісник Національного університету «Львівська політехніка». Серія: Радіоелектроніка та телекомунікації*. 2015. № 817. С. 102–106.

24. Al-Saka Y. A., Al-Saka T. A. Design and implementation of a universal microcontroller programmer. *International Journal of Computer Applications*. 2012. Т. 52, № 8. С. 1–6.

25. Kim K. J. Design and implementation of a universal programmer for microcontrollers using USB interface. *International Journal of Control, Automation, and Systems*. 2010. Т. 8, № 6. С. 1385–1392.

26. Chen R. S., Fu R. J., Yu C. L. A low cost universal microcontroller programmer with USB interface. *International Conference on Mechatronics and Automation (ICMA) : proceedings*. 2009. С. 2707–2711.

27. Huang H., Gu, Z. Research and implementation of AT89S52 microcontroller programmer. *International Conference on Electronic & Mechanical Engineering and Information Technology (EMEIT) : proceedings*. 2011. Т. 12. С. 6059–6062.

28. Kumar P., Sharma R. Design and implementation of AVR microcontroller programmer. *International Journal of Engineering and Technology*. 2016. Т. 8, № 3. С. 227–232.

29. Singh V. P., Kumar S. Development of a low cost microcontroller programmer for educational purposes. *International Journal of Engineering Research & Technology (IJERT)*. 2015. Т. 4, № 11. С. 367–370.

30. Sharma R. K., Kumar, S. Design of a universal microcontroller programmer. *International Journal of Scientific & Engineering Research*. 2014. Т. 5, № 5. С. 1374–1377.

31. Mahajan P. C., Bodhe S. K. A low cost Arduino based programmer for ATmega328P. *International Journal of Engineering Research and Technology (IJERT)*. 2019. Т. 8, № 5. С. 140–144.

32. Pathak S., Gupta N., Singh, R. Arduino based versatile microcontroller programmer. *International Journal of Scientific Research in Science and Technology*. 2019. Т. 5, № 5. С. 763–766.

33. Sharma A., Kumar A., Singh, B. P. Development of a low cost Arduino based microcontroller programmer. *International Journal of Engineering Science and Computing*. 2018. Т. 8, № 3. С. 16091–16094.

34. Agrawal A., Kumar V., Verma, S. K. Design and implementation of Arduino based universal programmer. *International Journal of Engineering Research and Technology (IJERT)*. 2019. Т. 8, № 2. С. 493–496.

35. Roy K., Das A. K., Roy D. Implementation of a low cost Arduino based microcontroller programmer. *International Journal of Recent Trends in Engineering & Research*. 2019. Т. 5, № 3. С. 49–53.

36. Gupta S., Singh R., Kumar A. Arduino based universal microcontroller programmer. *International Journal for Scientific Research & Development*. 2019. Т. 7, № 2. С. 83–86.

37. Yadav R. K., Verma R. P., Singh S. Design and development of a low cost Arduino based microcontroller programmer. *International Journal of Engineering Research & Technology (IJERT)*. 2019. Т. 8, № 4. С. 34–37.

38. Ali S. A., Khan M. I., Islam, R. Development of a low cost Arduino based universal programmer for ATmega microcontrollers. *International Journal of Advanced Research in Computer Science*. 2019. Т. 10, № 2. С. 284–288.

39. Khan A. K., Singh S., Kumar R. A low cost Arduino based universal programmer for microcontrollers. *International Journal of Scientific Research in Science and Technology*. 2019. Т. 5, № 4. С. 210–213.

40. Verma S., Garg A., Sharma R. Design and implementation of a low cost Arduino based microcontroller programmer. *International Journal of Engineering and Technology (UAE)*. 2019. Т. 8, № 2. С. 101–105.

41. Choudhary P., Gupta S., Kumar A. Design and implementation of a universal microcontroller programmer using Arduino. *International Journal of Engineering Science and Computing*. 2019. Т. 9, № 3. С. 21396–21399.

42. Mishra R., Kumar A., Dwivedi S. P. Arduino based universal microcontroller programmer. *International Journal of Recent Trends in Engineering & Research*. 2019. Т. 5, № 4. С. 14–18.

43. Sharma V., Singh Y., Kumar R. A low cost Arduino based universal microcontroller programmer. *International Journal of Engineering Research & Technology (IJERT)*. 2019. T. 8, № 5. C. 129–132.

44. Singh M. K., Kumar S. Development of a universal programmer for ATmega microcontroller using Arduino. *International Journal of Engineering Research and Technology (IJERT)*. 2019. T. 8, № 6. C. 34–37.

45. Sharma H., Kumar R. Design and implementation of Arduino based microcontroller programmer. *International Journal of Scientific Research in Science and Technology*. 2019. T. 5, № 3. C. 123–126.

46. Yadav R. K., Singh R. P., Kumar A. Low cost universal microcontroller programmer using Arduino. *International Journal of Engineering Research and Technology (IJERT)*. 2019. T. 8, № 7. C. 110–113.

47. Kumar S., Singh Y. K., Singh B. P. Arduino based universal programmer for ATmega series microcontrollers. *International Journal of Engineering Science and Computing*. 2019. T. 9, № 4. C. 21671–21674.

48. Gupta S., Singh R., Kumar A. Design and implementation of a low cost Arduino based microcontroller programmer. *International Journal of Scientific Research in Science and Technology*. 2019. T. 5, № 5. C. 789–792.

49. Pal N., Singh D. Arduino based universal programmer for 8051 and AVR microcontrollers. *International Journal of Advanced Research in Computer Science and Software Engineering*. 2019. T. 9, № 3. C. 75–78.

Додаток А

(обов’язковий)

ЛІТИНГ КОДУ ПРОГРАМИ

Лістинг коду програми

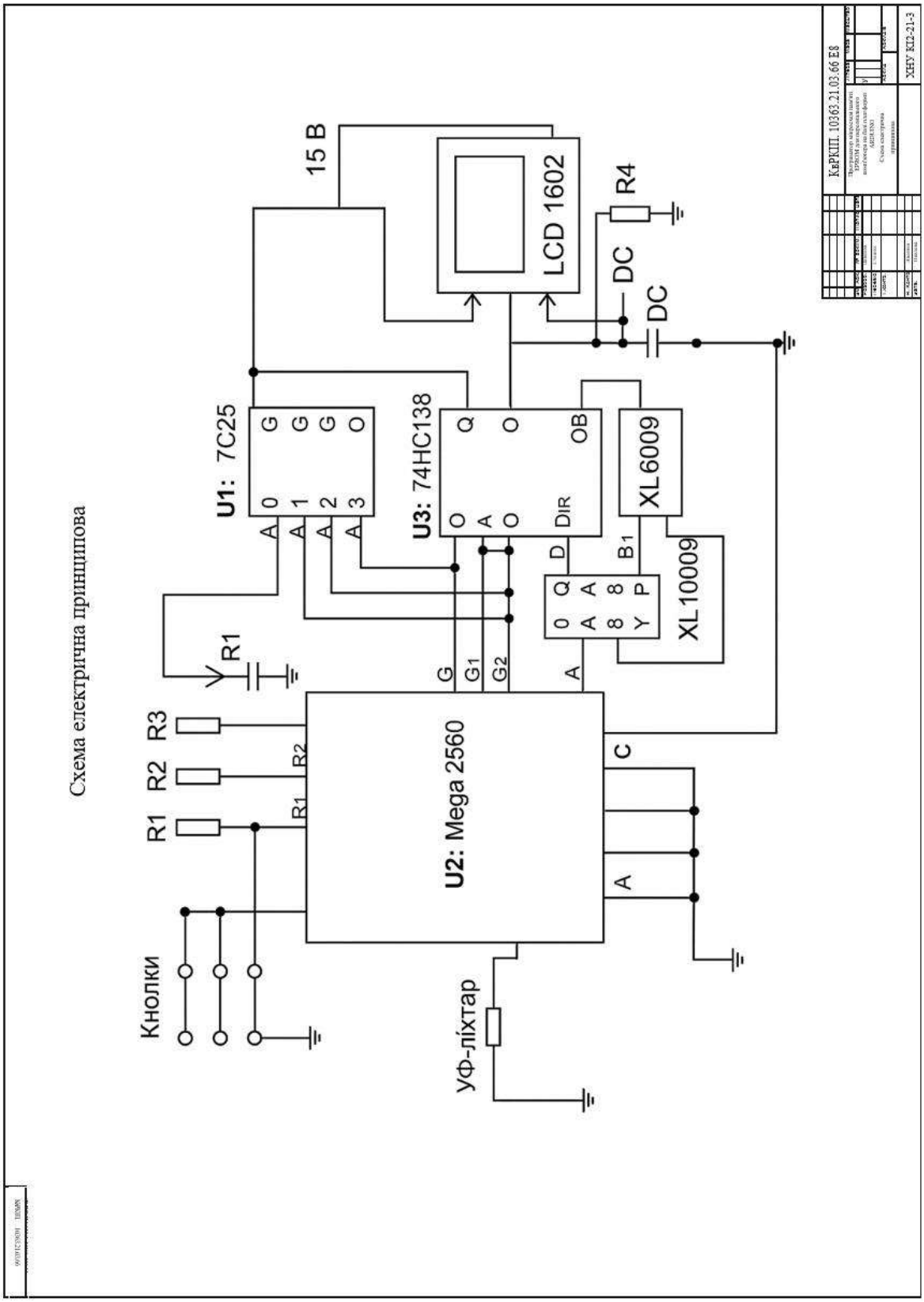
```
const int dataPins[] = {38, 39, 40,
41, 42, 43, 44, 45}; // D0-D7
const int addrPins[] = {22, 23, 24,
25, 26, 27, 28, 29, 30, 31, 32, 33,
34, 35, 36}; // A0-A14
const int CE = 46;     // Chip
Enable
const int OE = 47;     // Output
Enable
const int PGM = 48;    // Program
pin
const int VPP_CTRL = 52; //
Керування програмною напругою
void setupPins() {
  for (int i = 0; i < 15; i++)
pinMode(addrPins[i], OUTPUT);
  for (int i = 0; i < 8; i++)
pinMode(dataPins[i], OUTPUT);
  pinMode(CE, OUTPUT);
  pinMode(OE, OUTPUT);
  pinMode(PGM, OUTPUT);
  pinMode(VPP_CTRL, OUTPUT);
}
// Встановлення адреси в EPROM
void setAddress(unsigned int
address) {
  for (int i = 0; i < 15; i++) {
    digitalWrite(addrPins[i],
(address >> i) & 1);
  }
}
void setData(byte data) {
  for (int i = 0; i < 8; i++) {
    digitalWrite(dataPins[i],
(data >> i) & 1);
  }
}
// Запис одного байта у пам’ять
void programByte(unsigned int address,
byte data) {
  setAddress(address);
  setData(data);
  digitalWrite(CE, LOW);
  digitalWrite(PGM, LOW);
  digitalWrite(VPP_CTRL, HIGH); // Подача
високої напруги
  delayMicroseconds(50);
  digitalWrite(PGM, HIGH);
  digitalWrite(CE, HIGH);
  digitalWrite(VPP_CTRL, LOW);  //
Вимкнення програмної напруги
}
byte readByte(unsigned int address) {
  byte value = 0;
  setAddress(address);
  for (int i = 0; i < 8; i++)
pinMode(dataPins[i], INPUT);
  digitalWrite(OE, LOW);
  digitalWrite(CE, LOW);
  delayMicroseconds(10);
  for (int i = 0; i < 8; i++) {
    if (digitalRead(dataPins[i])) value |=
(1 << i);
  }
  digitalWrite(OE, HIGH);
  digitalWrite(CE, HIGH);
  for (int i = 0; i < 8; i++)
pinMode(dataPins[i], OUTPUT);
  return value;
}
// Увімкнення високої напруги
digitalWrite(VPP_CTRL, HIGH); // Увімкнення високої напруги
delayMicroseconds(50);
digitalWrite(PGM, HIGH);
digitalWrite(CE, HIGH);
digitalWrite(VPP_CTRL, LOW); // Вимкнення програмної
напруги
}
byte readByte(unsigned int address) {
  setAddress(address);
  for (int i = 0; i < 8; i++) pinMode(dataPins[i], INPUT);
  for (int i = 0; i < 8; i++) pinMode(dataPins[i], OUTPUT);
  return value;
}
```

| | | |

Додаток Б

(обов’язковий)

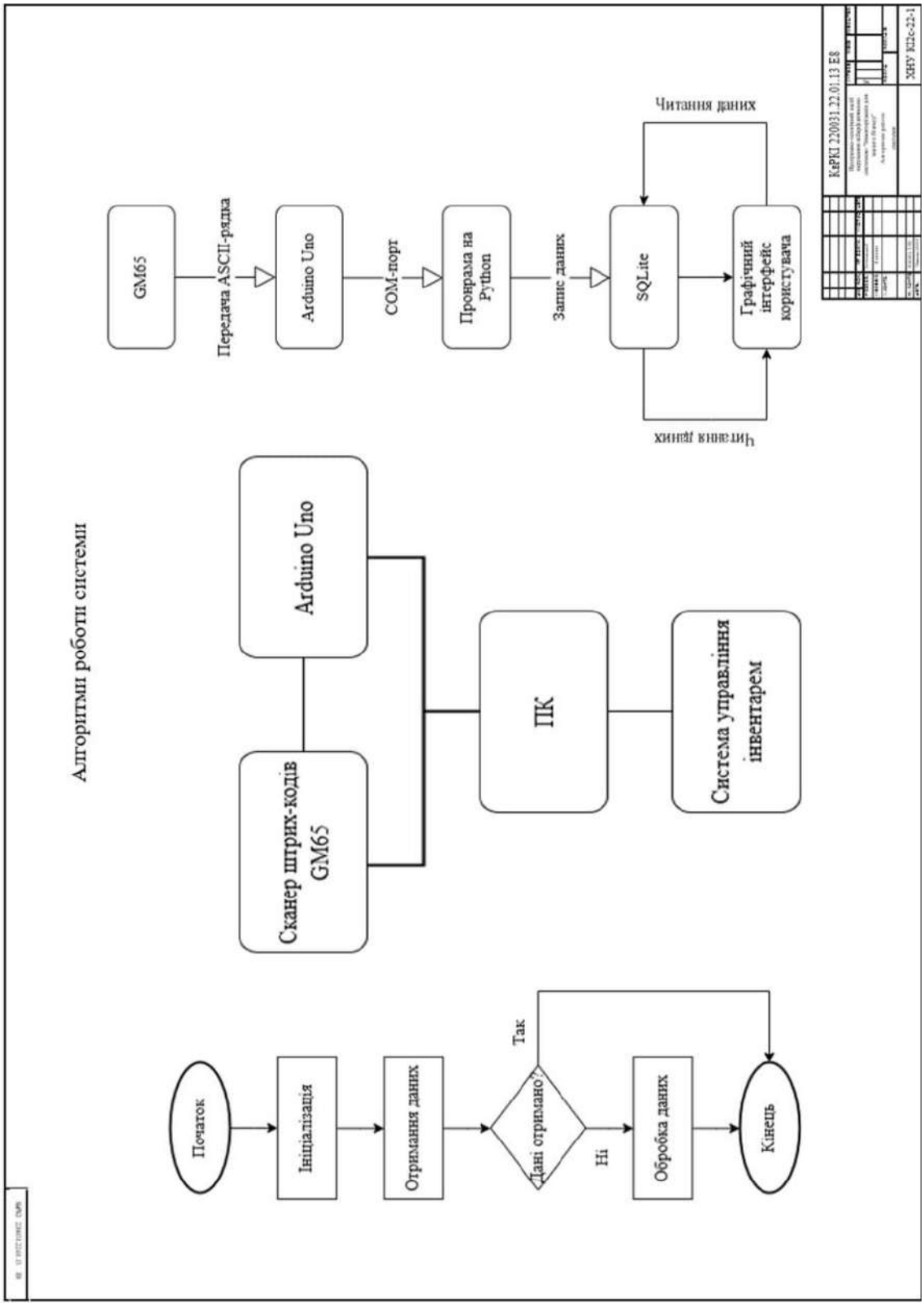
СХЕМА ЕЛЕКТРИЧНА ПРИНЦИПОВА



Додаток В

(обов’язковий)

АЛГОРИТМИ РОБОТИ СИСТЕМИ



Протокол аналізу звіту подібності експертом

Заявляю, що я ознайомився (-лась) з Повним звітом подібності, який був згенерований Системою виявлення і запобігання плагіату щодо роботи:

Автор: Василь СТЕЦЮК

Співавтор:

Назва: Давидов_Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino

Експерт:

Підрозділ: Кафедра комп'ютерної інженерії та інформаційних систем

Коефіцієнт подібності 1:5.9%

Коефіцієнт подібності 2:3.6%

Мікропробіли: 6

Заміна букв: 0

Інтервали: 0

Білі знаки: 0

Дата створення звіту: 2025-06-09 10:16:31.0

Після аналізу Звіту подібності констатую наступне:

☒ **Запозичення, виявлені в роботі є законними і не є плагіатом. Рівень подібності не перевищує допустимої межі. Таким чином робота незалежна і приймається.**

☐ **Запозичення не є плагіатом, але перевищено граничне значення рівня подібностей. Таким чином робота повертається на доопрацювання.**

☐ **Виявлено запозичення і плагіат або навмисні текстові спотворення (маніпуляції), як передбачувані спроби укриття плагіату, які роблять роботу невідповідною вимогам законодавства (Ст. 32. ЗУ Про вищу освіту, пункт 3.1, Ст. 42. ЗУ Про освіту) та вимог НАЗЯВО (Критерій 5), а також кодексу етики і процедурам. Таким чином робота не приймається.**

Обгрунтування:

2025-06-09 Доцент Андрій Нічепорук

Дата експерт

Anti-Plagiarism (UA) v-15.281 Educational

The maximum coincidence with one document 12.0%

Dictionaries check: en_US, ru_RU, ua_UA. **Errors in the documents: 10%**

ID: 244290 Title: БКР Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino Added in a DB: 2025-06-09 Authors: Дмитрій ДАВИДОВ Heads: Василь СТЕЦЮК Consultants: Opponents:	Document		Sum coincidence on the DB	
	Symbols	Lexemes	Symbols	Lexemes
	94608	640	15163 (16%)	115 (18%)

Plagiarism sources

ID	Description	Plagiarism presence in the document	
		Symbols	Lexemes
240773	Title: Звіт з ПДП Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи ARDUINO Added in a DB: 2025-05-02 Authors: Давидов Д.О. Heads: Стецюк В.М. Consultants: Opponents:	11527 (12.0%)	74 (12.0%)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

РЕЦЕНЗІЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ

Дипломник: Давидов Дмитрій Олегович

Тема: Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino

Спеціальність: 123 «Комп'ютерна інженерія»

Обсяг кваліфікаційної роботи:

Кількість листів креслень ___3____ Кількість сторінок записки ___57___

1. Короткий зміст роботи та прийнятих рішень: Метою кваліфікаційної роботи є розробка та реалізація пристрою - програматора мікросхем пам'яті EPROM

2. Висновок про відповідність роботи дипломному завданню: Робота повністю відповідає поставленому завданню.

3. Характеристика виконання кожного розділу, ступінь використання останніх досягнень науки і техніки і передових методів роботи: В першому розділі кваліфікаційної роботи сформовано чітке розуміння того, як повинна виглядати архітектура програматора, які елементи необхідно включити до схемотехнічної частини, які сигнали слід формувати для керування мікросхемою, а також які функції має реалізовувати програмна частина на платформі Arduino. У другому розділі дипломного проєкту було повністю реалізовано концепцію розробки програматора мікросхем EPROM на основі апаратної платформи Arduino, яка, завдяки своїй доступності, функціональності та простоті інтеграції, виявилася оптимальним вибором для побудови системи. Весь процес охопив як постановку задачі, так і побудову структурної, функціональної та алгоритмічної моделі, що забезпечило цілісність і завершеність розгляду. У третьому розділі підтвердив не лише функціональність створеної системи, а й її реальну придатність до експлуатації.

4. Позитивні сторони роботи: висока практична цінність роботи.

5. Негативні сторони роботи: відсутність інтегрованого модуля для автоматичного стирання мікросхем EPROM.

6. Оцінка графічного оформлення та пояснювальної записки роботи: Пояснювальна записка оформлена коректно, згідно діючих стандартів оформлення документації.

7. Відгук про роботу в цілому: Робота виконана на належному науково-технічному рівні.

8. Інші зауваження: _____

9. Оцінка дипломної роботи: задовільно

Рецензент (прізвище, ім'я, по батькові, посада, місце роботи) Мулир Ігор Володимирович, к.т.н., доцент кафедри кібербезпеки ХНУ

"12" 06 2025 р. _____(підпис)

Ігор МУЛЯР

Завідувачу кафедри КІІС
д-р. філософії, доц. Ользі ПАВЛОВІЙ

Дмитрія ДАВИДОВА
ПІБ здобувача вищої освіти

ФІТ, 4 курсу, групи КІ2-21-3

ЗАЯВА

З правилами чинного Положення «Про систему забезпечення академічної доброчесності у Хмельницькому національному університеті» від 01.07.2022, згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений (а). Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщений(а) та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

Також надаю університету право на передачу моєї роботи для обробки та збереження в базах даних програмно-технічних засобів (Strike-Plagiarism та Anti-Plagiarism) та використання роботи для виявлення плагіату в інших роботах, які перевіряються програмно-технічними засобами та користувачами, що мають доступ до цих програмно-технічних засобів, виключно в обмежених цілях для виявлення плагіату в текстах робіт.

Робота для перевірки університетом надається в друкованому та електронному варіанті. Електронна версія моєї роботи збігається (ідентична) з друкованою.

6 червня 2025 року

РІШЕННЯ ЕКСПЕРТНОЇ КОМІСІЇ

КАФЕДРИ КОМП'ЮТЕРНОЇ ІНЖЕНЕРІЇ ТА ІНФОРМАЦІЙНИХ СИСТЕМ

ПРО ДОПУСК КВАЛІФІКАЦІЙНОЇ РОБОТИ ДО ЗАХИСТУ

Підтверджуємо ознайомлення з результатом звіту подібності щодо роботи, генерованого системою виявлення текстових збігів/ідентичності/схожості:

Назва: Програматор мікросхем пам'яті EPROM для персонального комп'ютера на базі платформи Arduino

Автор: Дмитрій ДАВИДОВ

Спеціальність: 123– Комп'ютерна інженерія

Освітня програма: освітньо-професійна

Науковий керівник: Василь СТЕЦЮК, ст. викладач

Після аналізу звіту подібності зроблено такий висновок:

№	Висновок	Позначка про відповідність
1	Запозичення, виявлені в роботі, є законними і не є плагіатом. Робота приймається до захисту.	відповідає
2	Виявлені запозичення не є плагіатом, розміщені в розділах, які не описують безпосередньо авторське дослідження, але кількість цитат перевищує обсяг, виправданий поставленою метою роботи Робота приймається до захисту, але має бути відкоригована. Відкоригований варіант має бути поданий на кафедру за 2 дні до захисту, разом із заявою щодо самостійності виконання письмової роботи та ідентичності друкованої та електронної версії роботи	
3	Виявлені запозичення не є плагіатом, але частково розміщені в розділах, які описують безпосередньо авторське дослідження, а кількість цитат перевищує обсяг, виправданий поставленою метою роботи. В зв'язку з цим мета роботи та поставлені завдання не були досягнені. Робота може бути допущена до захисту (наступного року) після того як буде відкоригована та допрацьована і успішно пройде повторну перевірку на академічний плагіат.	
4	Робота містить навмисні текстові спотворення, передбачувані спроби укриття запозичень або інші прояви академічного плагіату. Робота містить фабрикацію або фальсифікацію даних. Робота не допускається до захисту.	

Підтвердження:

Запозичення, виявлені в роботі, є законними і не є плагіатом, оскільки:

1) запозичення розміщені в розділах аналізу існуючих аналогів та прототипів, які не описують безпосередньо авторське дослідження і не стосуються результатів роботи;
2) усі запозичення фрагментарні, або мають належним чином оформленні посилання;

Сумарний обсяг всіх запозичень, визначений системою виявлення збігів/ ідентичності/схожості StrikePlagiarism, складає 5,94% і адресується до 61 першоджерела; та системою Anti-Plagiarism складає 0,28%, що, з урахуванням наведених обґрунтувань, відповідає характеру наукового дослідження і свідчить на користь кваліфікаційної роботи.

Керівник роботи _____ Василь СТЕЦЮК

Гарант ОП _____ Андрій НІЧЕПОРУК

Завідувач кафедри КІІС _____ Ольга ПАВЛОВА