МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра електроніки і комп'ютерної техніки

«До захисту допущено»
Завідувач кафедри ЕКТ

_____ Анатолій ОПАНАСЮК
(підпис) (Ім'я та ПРІЗВИЩЕ)
____ _____ 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА

на здобуття освітнього ступеня «магістр»

зі спеціальності 171 «Електроніка»

освітньо-професійної програми «Електронні системи»

на тему:

ЕЛЕКТРОННА СИСТЕМА ЗАХИСТУ ІНФОРМАЦІЇ НА БАЗІ МОНОАЛФАВІТНОГО АЛГОРИТМУ ШИФРУВАННЯ

Здобувача групи ЕС.м-31 Мельника Романа Валерійовича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Мельника Романа
(підпис) (Ім'я та ПРІЗВИЩЕ)

Керівник, доцент, к.т.н., доцент Ольга БЕРЕЖНА _____
(підпис)

Консультант з техніко-економічної частини,
доцент, к.е.н., доцент Олександр МАЦЕНКО _____
(підпис)

Суми – 2024

СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ

Факультет електроніки та інформаційних технологій

Кафедра електроніки і комп'ютерної техніки

Напрям підготовки 171 «Електроніка»

Освітня програма Електронні системи

ЗАТВЕРДЖУЮ

Зав. кафедрою Опанасюк А. С.

«___»_____ 2024 р.

З А В Д А Н Н Я

на кваліфікаційну роботу магістра

Мельнику Роману Валерійовичу

1. Тема роботи «Електронна система захисту інформації на базі моноалфавітного алгоритму шифрування».
затверджена наказом по університету «01» жовтня 2024 р. № 1003-VI.

2. Термін здачі студентом завершеної роботи 05.12.2024.

3. Вихідні дані до роботи Розробити електронну систему захисту інформації на базі моноалфавітного алгоритму шифрування з використанням шифру Цезаря з ключовим словом для мультиалфавітних джерел інформації. Принципову схему реалізувати із застосуванням мікроконтролеру.

4. Зміст розрахунково-пояснювальної записки (перелік питань, що належить розробити) 1) Огляд літератури та поставлення задачі роботи. 2) Науково-дослідна частина. 3) Розробка електронної системи з використанням отриманих результатів дослідження. 4) Техніко-економічна частина.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) 1) Схема електрична структурна. 2) Схема алгоритму. 3) Схема електрична функціональна. 4) Схема електрична принципова.

6. Консультанти з кваліфікаційної роботи

Розділи	Консультанти	Завдання видав	Завдання прийняв
Техніко-економічна частина	Маценко О. М.		

7. Дата видачі завдання _____

8. Керівник роботи _____

9. Завдання прийняв до виконання _____

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів дипломного проекту	Термін виконання етапів роботи	Примітки
1	Огляд літератури та постановка завдання проектування	04.11.24 – 09.11.24	
2	Науково-дослідна частина	10.11.24 – 15.11.24	
3	Розробка алгоритму функціонування та структурної схеми електронної системи	16.11.24 – 20.11.24	
4	Розробка функціональної схеми електронної системи	21.11.24 – 24.12.24	
5	Розробка схеми електричної принципової електронної системи	25.12.24 – 02.12.24	
6	Техніко-економічна частина	03.12.24 – 05.12.24	
8	Оформлення пояснювальної записки	06.12.24 – 08.12.24	
9	Оформлення графічного матеріалу	09.12.24 – 13.12.24	
10	Представлення роботи керівнику і отримання відгуку	14.12.24	
11	Представлення роботи кафедрі для отримання рецензії	15.12.24	

Студент _____

Керівник роботи _____

«___» _____ 2024 р.

РЕФЕРАТ

Записка: 88 сторінок, 38 рисунків, 9 таблиць, 12 джерел.

Тема роботи: «Електронна система захисту інформації на базі моноалфавітного алгоритму шифрування».

Об'єктом розробки є електронна система захисту інформації на базі моноалфавітного алгоритму шифрування.

Мета роботи – розробка апаратної частини кодуючої електронної системи.

Пояснювальна записка складається з шести розділів.

У першому розділі наданий огляд алгоритмів, досліджень та порівняння моноалфавітних шифрів.

У другому розділі проводиться оцінка стійкості шифруючих перетворень моноалфавітної заміни з використанням генетичного алгоритму.

У третьому розділі проводиться розробка структурної схеми. Описані блоки структурної електричної схеми.

У четвертому розділі проводиться розробка функціональної схеми електронної системи.

У п’ятому розділі проведена розробка принципової схеми проектованої електронної системи. Також показана програма, за якою електронна система працює.

У шостому розділі представлено схему розрахунку собівартості проектованої електронної системи при серійному її виробництві.

У висновках наводяться результати розробки електронної системи.

Ключові слова: електронна система, моноалфавітний алгоритм, шифрування, мікропроцесор.

ЗМІСТ

ВСТУП

Захист інформації став одним з пріоритетів у сучасному світі, де більшість даних передається, обробляється і зберігається в електронному вигляді. Кожна установа, компанія чи окрема особа стикається з потребою захисту своїх конфіденційних даних від несанкціонованого доступу, викрадення або маніпуляцій. Використання шифрування як засобу захисту даних дозволяє перетворювати інформацію у формат, недоступний для сторонніх осіб, що особливо важливо в умовах сучасних кіберзагроз. Одним з найпростіших та базових методів є моноалфавітний алгоритм шифрування, який передбачає заміну символів за певним ключем. Хоча моноалфавітні шифри мають обмежену стійкість до криптоаналізу, вони є відносно швидкими, економічними і можуть бути адаптовані для завдань, що не вимагають високого рівня захисту.

Моноалфавітний алгоритм шифрування є актуальним для застосування у системах з обмеженими ресурсами, таких як IoT-пристрої або індивідуальні програми захисту персональних даних. Також цей метод корисний у випадках, коли пріоритетним є саме швидке виконання шифрування, а не його надвисока складність. Розробка електронної системи захисту інформації, заснованої на моноалфавітному шифруванні, дозволяє створити легке та ефективне рішення для базового рівня захисту, яке можна впроваджувати в різних сферах. Це дослідження має практичне значення для малого бізнесу, особистих користувачів та установ, що стикаються з проблемами захисту внутрішніх даних.

1 ОГЛЯД ЛІТЕРАТУРИ ТА ПОСТАНОВКА ЗАВДАННЯ

1.1 Розвиток шифрування та історія моноалфавітних алгоритмів

Шифрування як метод забезпечення конфіденційності інформації виникло тисячі років тому. Спочатку його використовували для приховування повідомлень військового чи політичного характеру. Одним із найбільш відомих і найраніших прикладів такого шифрування є шифр Цезаря, де кожна літера тексту замінюється на іншу, зміщену на певну кількість позицій у алфавіті. Цей шифр є представником класу моноалфавітних алгоритмів, що базуються на використанні одного і того ж алфавіту заміни для всього тексту. Його простота зробила його дуже популярним у свій час, але також і вразливим до криптоаналітичних методів.

Моноалфавітні шифри [5] використовують принцип одноразової заміни символів з одного алфавіту на символи іншого, що може забезпечувати базовий рівень приховування інформації. У таких шифрах кожен символ має фіксовану відповідність у шифрованому тексті, що дозволяє передавати повідомлення без їхнього явного викриття. Однак цей принцип має суттєві недоліки з точки зору криптографії, оскільки регулярність замін робить їх передбачуваними. Моноалфавітні шифри не здатні забезпечити захист від складніших атак, таких як частотний аналіз, оскільки вони не змінюють розподіл частоти символів, що дозволяє потенційному зловмиснику аналізувати шифротекст.

Зростання обізнаності щодо обмежень моноалфавітних шифрів призвело до виникнення поліалфавітних методів, таких як шифр Віженера, які використовують змінні алфавіти для кожної частини тексту. Цей підхід дозволив значно посилити захист інформації, оскільки зміна алфавітів значно ускладнює розшифрування, зокрема частотний аналіз. Поліалфавітні шифри стали значним кроком уперед у криптографії, особливо для військових та дипломатичних цілей, де безпека була критично важливою. Проте моноалфавітні шифри не зникли повністю – вони збереглися як навчальний інструмент для ілюстрації основ криптографії.

Моноалфавітні шифри мають не лише історичну, але й освітню цінність, оскільки вони демонструють початкові принципи шифрування. Науковці продовжують використовувати їх як модель для вивчення основ шифрування, а також аналізу базових криптоаналітичних методів. Розуміння історії шифрування,

зокрема моноалфавітних шифрів, дозволяє оцінити переваги сучасних методів та підходів до шифрування, які є складнішими і надійнішими [11].

1.2 Вивчення криптостійкості моноалфавітних шифрів

Частотний аналіз [4] , один із головних методів криптоаналітичних атак, був розроблений ще в епоху Відродження. Він дозволяв розкривати моноалфавітні шифри завдяки статистичному аналізу частот появи різних символів. Наприклад, англійська літера «E» є однією з найпоширеніших у текстах, тому, якщо в зашифрованому повідомленні певний символ зустрічається найчастіше, можна припустити, що він відповідає букві «E». Аналіз частоти символів дозволяє знаходити закономірності, які полегшують розшифрування тексту навіть без знання ключа.

Через вразливість до частотного аналізу моноалфавітні шифри стали поступово втрачати популярність у практичному застосуванні, оскільки криптоаналітики розвинули методи аналізу текстів, засновані на математичних та статистичних закономірностях. Ранні криптоаналітики вивчали мову та алфавітні патерни, що дозволяло їм виявляти слабкі місця у шифрованих повідомленнях. Це призвело до того, що криптографи почали шукати альтернативні методи для підвищення стійкості моноалфавітних шифрів або переходили на складніші алгоритми.

Дослідники також вивчали способи модифікації моноалфавітних шифрів для підвищення їх криптостійкості. Наприклад, у деяких випадках пропонувалося замінювати найчастіше вживані символи на менш поширені або використовувати спеціальні коди для їх приховування. Проте такі методи були неефективними, оскільки не могли повністю усунути вразливість до частотного аналізу. Це підштовхнуло криптографію до пошуку більш ефективних рішень, які змогли б забезпечити надійний захист даних.

Згодом, для підвищення безпеки, криптографія почала переходити до поліалфавітних шифрів і складніших алгоритмів, які дозволяють знизити вразливість до криптоаналітичних атак. Проте моноалфавітні шифри залишаються важливим етапом у розвитку криптографії, оскільки вони заклали основи для розуміння більш складних методів. Тому, хоча сучасні шифри значно

перевершують моноалфавітні за безпекою, останні продовжують залишатися цінним інструментом для навчання основам шифрування.

1.3 Сучасні дослідження та методи підвищення криптостійкості

Сьогодні науковці продовжують досліджувати способи підвищення стійкості моноалфавітних шифрів, хоча вони й не розглядаються як основний метод для захисту даних. Один із сучасних підходів включає динамічну зміну алфавіту заміни протягом шифрування тексту. Такий підхід може ускладнити частотний аналіз, оскільки змінний ключ порушує традиційні закономірності в розподілі частот, тим самим зменшуючи ефективність атак.

Іншим методом є поєднання моноалфавітного шифру з додатковими криптографічними операціями, наприклад, обробкою тексту за допомогою хеш-функцій або перетворень, які змінюють порядок символів. Цей метод забезпечує маскування зв'язків між початковим і зашифрованим текстом, що ускладнює можливість відновлення оригінального повідомлення. Деякі дослідження показують, що такі комбіновані методи можуть досягати кращих результатів у стійкості до атак, ніж прості моноалфавітні шифри.

У контексті пристроїв з обмеженими ресурсами, таких як сенсори та вбудовані системи, моноалфавітні шифри все ще розглядаються як можливе рішення, оскільки вони мають низькі обчислювальні вимоги. Простота моноалфавітних шифрів дозволяє їх ефективно використовувати там, де високий рівень безпеки не є пріоритетом, але потрібен певний базовий захист даних. Використання цих методів у поєднанні з іншими захисними засобами може забезпечити задовільний рівень безпеки в умовах обмежених ресурсів.

Незважаючи на ці спроби, моноалфавітні шифри не є конкурентоспроможними з сучасними методами шифрування, такими як AES чи RSA, які надають набагато вищий рівень безпеки. Проте дослідження методів покращення моноалфавітних шифрів продовжуються, що дозволяє зрозуміти обмеження і можливості цих алгоритмів для спеціалізованих застосувань.

1.4 Аналіз сучасних досягнень у сфері моноалфавітного шифрування

Останні дослідження, пропонують комбіновані підходи, в яких моноалфавітний шифр інтегрується з іншими криптографічними методами, такими як поліалфавітне шифрування або шифри перестановки. Ці підходи створюють складніші шифри, що можуть краще протидіяти простим атакам, зокрема частотному аналізу. Наприклад, комбінація моноалфавітного шифру з перестановкою символів у тексті порушує закономірності, що ускладнює розпізнавання структури повідомлення.

Сучасні криптографічні дослідження зосереджуються на тому, як поєднувати простоту моноалфавітного шифру з більш захищеними методами, щоб забезпечити обмежену стійкість до атак. Крім того, деякі науковці вивчають можливість динамічного оновлення ключа під час передачі даних, що ускладнює злом через відсутність постійної закономірності у розподілі символів. Такі методи є особливо актуальними для сенсорних мереж, де надмірно складні шифри можуть уповільнювати роботу.

Моноалфавітні шифри мають значення, навіть якщо їхня актуальність для захисту даних поступово зменшується. У спеціалізованих випадках вони все ще можуть бути використані, а також залишаються цінним ресурсом для розуміння історії та основних принципів криптографії.

1.5 Симетричні криптосистеми

Усі криптографічні системи поділяються на два класи: симетричні криптографічні системи [2] та асиметричні криптографічні системи з відкритим ключем [3].

У симетричних криптосистемах при шифруванні та дешифруванні використовують секретний ключ. Важливим моментом тут є те, що функції шифрування та дешифрування відкриті, а секретний ключ закритий і від його секретності залежить секретність всього алгоритму шифрування.

Секретність симетричної криптосистеми повністю залежить від безпеки ключа, оскільки функції шифрування та дешифрування є відкритими. Якщо зловмисник отримає доступ до ключа, він зможе дешифрувати всі повідомлення. Саме тому важливим аспектом є захищений канал для передачі ключа.

До основних переваг симетричних криптосистем належать їхня висока швидкість роботи та ефективність. Це особливо важливо при шифруванні великих обсягів даних. Основними недоліками є складність управління ключами, оскільки для кожної пари користувачів потрібен окремий ключ, і необхідність безпечного обміну цими ключами.

Роботу симетричної криптосистеми можна описати в такий спосіб: Перший користувач збирається надіслати другому користувачу зашифроване повідомлення (відкритий текст m). Для цього перший користувач шифрує повідомлення за допомогою секретного ключа k. Отримана в результаті шифрограма передається по відкритому каналу зв’язку.

Другий користувач отримавши шифрограму, розшифровує її за допомогою того ж секретного ключа. Користувач 1 та 2 мають один і той же секретний ключ. Вважається, що користувачі отримують ключ від певного джерела ключів.

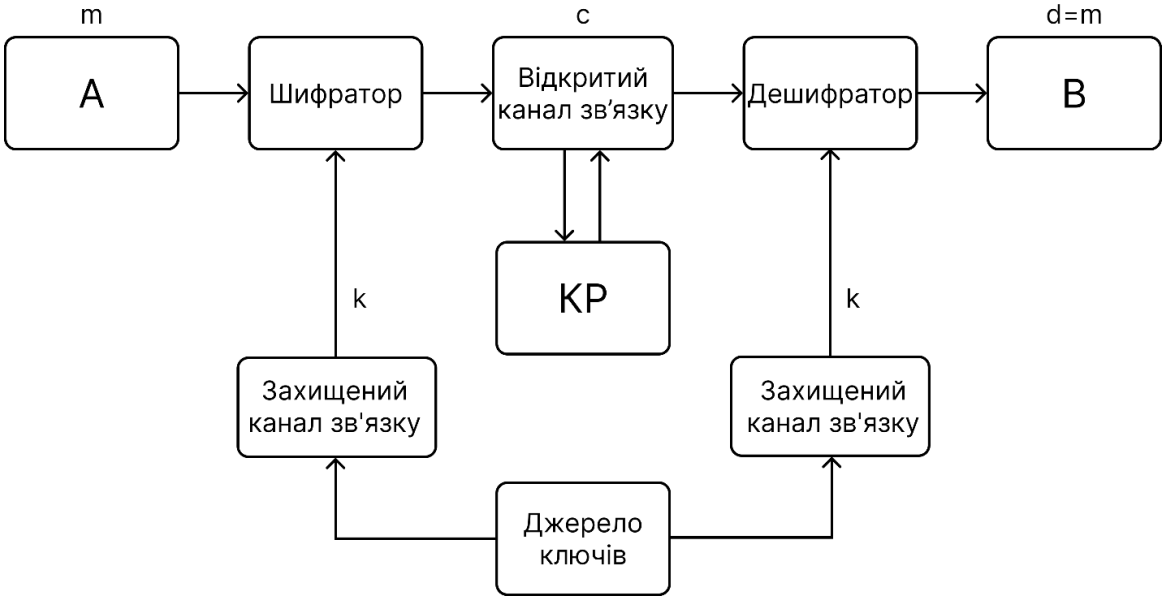


Рисунок 1.1 – Блок схема симетричної криптосистеми

Символами A та B позначають користувача 1 та користувача 2

m - відкритий текст.

c - шифротекст (шифрограма, криптограма)

d - розшифрований текст

k - секретний ключ

1.6 Обґрунтування вибору моноалфавітного алгоритму

Моноалфавітний алгоритм, незважаючи на його вразливість до частотного аналізу, має свої переваги, що робить його привабливим вибором для окремих завдань. Серед основних переваг – простота реалізації та мінімальні вимоги до ресурсів, що дозволяє застосовувати його у випадках, де потрібен лише базовий рівень захисту інформації. У низці сценаріїв забезпечення простоти алгоритму є ключовим фактором, особливо якщо йдеться про тимчасові системи з низькими вимогами до стійкості. Моноалфавітний алгоритм дозволяє швидко й ефективно шифрувати дані з мінімальним впливом на продуктивність системи.

Ще одним фактором, що вплинув на вибір цього алгоритму, є його придатність для використання в умовах обмежених обчислювальних ресурсів. Високі вимоги до захисту не завжди є необхідними, наприклад, у системах локальної безпеки, де ризик складних атак залишається низьким. У таких випадках простота та швидкість моноалфавітного шифру стають важливішими за криптостійкість. Завдяки відсутності складних математичних операцій цей шифр споживає значно менше енергії та пам'яті.

Також важливо врахувати, що моноалфавітний шифр може бути ефективним у випадках, коли дані швидко втрачають актуальність. Для тимчасової або частково захищеної інформації він забезпечує достатній рівень конфіденційності, який відповідає вимогам до таких систем. Швидке шифрування та дешифрування дозволяє реалізувати цей алгоритм у пристроях, де потрібна базова прихованість без обмеження продуктивності системи.

Узагальнюючи, вибір моноалфавітного алгоритму ґрунтується на потребі забезпечити компроміс між ефективністю, швидкістю та низькими ресурсними вимогами. Простота реалізації робить його привабливим варіантом для обмежених середовищ, у яких складніші алгоритми є надмірними або недоступними через високі обчислювальні витрати. Цей шифр дозволяє підтримувати мінімальний рівень захисту в системах, де складні методи є непрактичними або недоцільними [9].

1.7 Переваги простоти та швидкості реалізації

Моноалфавітний алгоритм є одним із найбільш простих у реалізації, що значно полегшує інтеграцію в різні системи. Його простота дозволяє швидко налаштувати алгоритм для роботи навіть у обмежених пристроях, де є дефіцит апаратних ресурсів. Це забезпечує швидке шифрування та дешифрування, що може бути важливим фактором у системах, які працюють у режимі реального часу. Простота реалізації також означає, що навіть недосвідчені розробники можуть налаштувати та інтегрувати алгоритм з мінімальними витратами.

Швидкість роботи моноалфавітного алгоритму є ще одним важливим фактором для його вибору, оскільки він не вимагає великих обчислювальних ресурсів. У середовищах з обмеженими обчислювальними потужностями швидкість шифрування дозволяє обробляти дані в реальному часі, не обмежуючи роботу основної системи. Це є особливо корисним для систем, де час обробки має критичне значення, наприклад, у системах збору даних або контролю доступу.

Крім того, завдяки відсутності складних математичних обчислень моноалфавітний алгоритм можна легко адаптувати для пристроїв з обмеженими ресурсами, такими як мікроконтролери або сенсори. Це робить його популярним у пристроях IoT, де важливо зберігати невеликі енергоспоживання та обмежену потужність обчислень. Таким чином, алгоритм забезпечує баланс між шифруванням і продуктивністю системи, не знижуючи її ефективності.

Загалом, простота та швидкість є ключовими перевагами цього алгоритму, що дозволяє використовувати його для широкого кола завдань, де високий рівень безпеки не є обов’язковим. Зниження вимог до ресурсів робить його доступним для вбудованих систем, дозволяючи розробникам зменшити витрати на розробку і підтримку системи.

1.8 Вибір специфічних сценаріїв застосування

Моноалфавітний алгоритм залишається актуальним для певних сфер застосування, зокрема для пристроїв і систем із низьким рівнем ризику складних атак. Наприклад, у сенсорних мережах або пристроях Інтернету речей (IoT), де передаються дані, що швидко втрачають актуальність, простий алгоритм шифрування може забезпечити базовий рівень конфіденційності. У таких

системах пріоритетом є обмеження витрат на обчислення, а не максимальний захист інформації.

Деякі інформаційні системи використовують моноалфавітний алгоритм для шифрування даних, доступних лише локально або в обмеженому середовищі, де доступ третіх осіб виключений. Наприклад, дані, що зберігаються на спеціалізованих пристроях, можуть бути зашифровані для запобігання випадковому доступу до них без необхідності ускладнювати алгоритм. Це допомагає забезпечити базовий захист без додаткових витрат на інфраструктуру безпеки.

У деяких випадках, таких як тимчасове зберігання інформації, моноалфавітний шифр може виконувати роль бар'єру від неавторизованого доступу. Наприклад, він може використовуватися для захисту тимчасових даних у системах збору інформації або контролю доступу, де потрібен лише мінімальний рівень захисту. Це дозволяє уникнути складних рішень, зберігаючи необхідний рівень швидкості роботи.

Вибір моноалфавітного алгоритму в специфічних сценаріях обґрунтований його простотою, швидкістю та мінімальними витратами на реалізацію. Він є практичним вибором для тимчасового або часткового захисту інформації у випадках, коли повна криптостійкість не є критично важливою.

1.9 Мінімізація витрат ресурсів та спрощення системи

Використання моноалфавітного алгоритму дозволяє значно зменшити витрати на розробку та підтримку системи захисту інформації. На відміну від складних криптографічних рішень, що потребують значних ресурсів для впровадження та підтримки, моноалфавітні шифри мають мінімальні вимоги до інфраструктури. Це дозволяє знизити загальні витрати, що є важливим чинником для проєктів з обмеженим бюджетом або для пристроїв, які потребують дешевих і простих рішень.

Крім того, спрощення архітектури шифрувальної системи дає змогу скоротити вимоги до обчислювальних ресурсів, що сприяє підвищенню ефективності. Зниження навантаження на процесор і пам'ять дозволяє інтегрувати шифр у різноманітні вбудовані пристрої та системи контролю доступу. Це особливо важливо для пристроїв з обмеженими можливостями, де висока

криптостійкість може бути надмірною, тоді як простий шифр буде цілком достатнім.

Мінімізація вимог до ресурсів є вагомою перевагою для вбудованих систем, де важливо забезпечити базовий рівень захисту, не збільшуючи навантаження на обладнання. Спрощення структури шифру сприяє ефективній інтеграції в систему та знижує ризик виникнення помилок під час налаштування та експлуатації. Це допомагає забезпечити стабільність роботи всієї системи, зменшуючи ризики збоїв.

Моноалфавітний алгоритм дозволяє досягти балансу між простотою та функціональністю, що робить його вигідним вибором у випадках, коли складні рішення є непотрібними або недоступними. Він надає можливість реалізувати систему захисту з мінімальними витратами ресурсів, зберігаючи при цьому базовий рівень безпеки.

1.10 Загальна методика дослідження

Методика дослідження охоплює комплексний підхід, що включає аналіз теоретичних основ і порівняння моноалфавітного алгоритму з іншими методами захисту інформації. Початковий етап полягає у вивченні основних положень криптографії, зокрема принципів роботи класичних і сучасних алгоритмів шифрування, а також вимог до інформаційної безпеки. Це дозволяє визначити, наскільки моноалфавітний алгоритм може відповідати сучасним викликам і які переваги він має у певних випадках.

Особливу увагу приділено порівнянню моноалфавітного алгоритму з іншими, більш сучасними методами захисту, такими як поліалфавітні та симетричні шифри. Порівняння проводиться на основі аналізу ключових параметрів, таких як швидкість обробки, вимоги до ресурсів, а також криптостійкість. Це допомагає обґрунтувати доцільність вибору моноалфавітного шифру для конкретних сценаріїв, де вимоги до стійкості є помірними, але потрібна простота реалізації.

1.11 Порівняння моноалфавітних шифрів з сучасними методами шифрування

Моноалфавітні шифри, такі як шифр Цезаря, шифр Атбаш та шифр простої заміни, є основою криптографії та володіють певними перевагами в специфічних

випадках. Проте їхня криптографічна стійкість є дуже низькою в порівнянні з більш сучасними методами, такими як поліалфавітні шифри (наприклад, шифр Віженера) та симетричні шифри (AES, DES тощо). Для кращого розуміння сильних і слабких сторін моноалфавітних шифрів у порівнянні з новими підходами до шифрування, давайте розглянемо їх у контексті таких ключових параметрів, як швидкість обробки, вимоги до ресурсів і криптостійкість.

Моноалфавітні шифри є надзвичайно швидкими в процесі шифрування та розшифрування. Вони здійснюють просту заміну кожного символу тексту, що не вимагає складних математичних операцій.

Для більшості моноалфавітних шифрів достатньо лише один раз перевірити відповідність символів у алфавіті (у випадку Цезаря – це просто зсув на певну кількість позицій), що робить їх дуже ефективними для швидкої обробки навіть великих обсягів даних. Шифрування чи розшифрування може бути виконано вручну або на базі елементарних програм, що дозволяє швидко отримувати зашифровані повідомлення.

Поліалфавітні шифри [6] (наприклад, шифр Віженера) теж є швидкими, але їхня складність більша, оскільки кожен символ тексту може бути зашифрований різними способами залежно від ключа. Це вимагає більших обчислювальних зусиль порівняно з моноалфавітними шифрами, хоча на сучасних обчислювальних системах ця різниця є незначною для звичайних обсягів даних. Зокрема, поліалфавітні шифри можуть використовувати таблиці для швидкого доступу до зворотних перетворень.

Симетричні шифри, такі як AES, є значно складнішими і вимагають більше ресурсів для шифрування та розшифрування. Вони працюють на основі складних математичних операцій (наприклад, змішування, підстановка, перестановки), що збільшує час обробки. Однак на сучасних апаратних платформах, зокрема за допомогою апаратних прискорювачів, швидкість роботи симетричних шифрів також є дуже високою, і вона не має суттєвого впливу на ефективність обробки даних.

Моноалфавітні шифри мають мінімальні вимоги до ресурсів. Вони не потребують складних обчислень, великих обсягів пам'яті або високої потужності процесора. Більшість з них може бути реалізована вручну або за допомогою базових комп'ютерних програм, що працюють на простих пристроях, таких як калькулятори або навіть на папері. Це робить їх ідеальними для використання в

умовах обмежених ресурсів або коли немає доступу до складних обчислювальних потужностей.

Поліалфавітні шифри, хоча і вимагають більше ресурсів для реалізації, можуть бути також ефективно впроваджені навіть на простих комп'ютерах. Для реалізації шифру Віженера або інших поліалфавітних методів достатньо мати таблицю алфавітів і базові функції для обчислення зсувів або заміни. Вони займають більше пам'яті та часу, ніж моноалфавітні шифри, але все одно можуть бути виконані навіть на пристроях з обмеженими ресурсами.

Симетричні шифри, зокрема AES, потребують значно більше ресурсів для виконання шифрування та розшифрування. Вони вимагають як мінімум 128 біт для ключа, а також великих обсягів пам'яті та процесорних потужностей для реалізації складних операцій заміни та змішування. На старших або менш потужних пристроях ці шифри можуть бути менш ефективними і потребувати додаткових апаратних ресурсів, таких як криптографічні прискорювачі.

Криптографічна стійкість моноалфавітних шифрів є дуже низькою. Оскільки кожна буква замінюється на одну й ту саму букву, шифр має велику вразливість до частотного аналізу. Відомо, що в більшості природних мов деякі літери (наприклад, у англійській "E", "T", "A") з'являються значно частіше за інші, і тому можна розпізнати, які літери шифруються однією і тією ж заміною. Це дозволяє легко зламати шифр методом частотного аналізу навіть за короткий час, особливо якщо текст великий.

Поліалфавітні шифри значно складніші для криптоаналізу, оскільки кожен символ шифрується за допомогою різних замін, що значно ускладнює частотний аналіз. Наприклад, у шифрі Віженера кожен символ тексту має свій ключ, і це дозволяє ускладнити відновлення оригінального тексту. Однак поліалфавітні шифри все ще уразливі до більш складних атак, таких як криптоаналіз Касіса або аналіз з використанням статистичних методів.

Симетричні шифри, такі як AES, мають високу криптографічну стійкість. Вони використовують великі ключі (наприклад, 128, 192 або 256 біт) і складні математичні операції, що робить їх дуже важкими для зламування навіть при використанні сучасних обчислювальних потужностей. Крім того, ці шифри стійкі до всіх відомих методів криптоаналізу, включаючи частотний аналіз і методи на основі статистики.

Моноалфавітні шифри можуть бути вибрані для використання в певних ситуаціях, де важливими є такі фактори, як простота реалізації, швидкість роботи та мінімальні вимоги до ресурсів. Вони можуть бути доцільними в умовах, коли важливість безпеки не є першочерговою, або коли криптографічний захист необхідний для менш чутливих даних. Наприклад, моноалфавітний шифр, такий як шифр Цезаря або Атбаш, може бути вибраний для цілей, де важливо швидко зашифрувати або розшифрувати невеликі обсяги інформації, наприклад, для персональних повідомлень або простих ігор.

У реальних умовах моноалфавітні шифри можуть бути корисними, коли потрібно передати інформацію в умовах обмежених ресурсів, де доступ до складних криптографічних засобів обмежений.

Моноалфавітний шифр може бути вибраний для внутрішнього використання в організаціях, де важливість конфіденційності є меншою, а завдання зберігання приватності не є критичними. Наприклад, для тимчасових зашифрованих повідомлень між співробітниками, де головним є швидкість і простота, а не високий рівень безпеки. У таких випадках моноалфавітні шифри зручно використовувати, оскільки вони не потребують складних обчислень і можуть бути легко реалізовані навіть за допомогою ручного процесу.

У реальності моноалфавітні шифри, незважаючи на їхню низьку криптографічну стійкість, можуть використовуватися для завдань, які не вимагають високого рівня захисту, але при цьому важливо, щоб метод був швидким, простим і зрозумілим [10].

1.12 Моноалфавітні шифри

1.12.1 Шифр Цезаря

Шифр Цезаря [1] або шифр зсуву – симетричний моноалфавітний алгоритм шифрування, в якому кожна буква відкритого тексту заміняється на ту, що віддалена на неї в алфавіті на сталу кількість позицій.

Принцип дії полягає в тому, щоб циклічно зсунути алфавіт, а ключ — це кількість літер, на які робиться зсув.

Наприклад для літери А (в українському алфавіті) при ключі 5:

А→Б→В→Г→Ґ→Д

Якщо після зміщення літера виходить за межі алфавіту, вона повертається до початку.

Наприклад для Я при ключі 3:

Я→А→Б→**В**

Якщо зіставити кожному символу алфавіту його порядковий номер (нумеруючи з 0), то шифрування і дешифрування можна виразити формулами:

$$y = (x + k) mod\ n, \tag{1.1}$$

$$x = (y - k) mod\ \text{n}, \tag{1.2}$$

де x — порядковий номер символу відкритого тексту, y — порядковий номер символу шифрованого тексту, n — кількість символів в алфавіті, а k — ключ.

Приклад шифрування. Візьмемо слово МАГІСТР з ключом зсуву 4 .

Застосовуємо зсув на 4:

М→Н→О→П→Р
А→Б→В→Г→Ґ
Г→Ґ→Д→Е→Є
І→Ї→Й→К→Л
С→Т→У→Ф→Х
Т→У→Ф→Х→Ц
Р→С→Т→У→Ф

Рисунок 1.2 – Зсув за алгоритмом Цезаря

Шифроване слово - РҐЄЛХЦФ

Шифр Цезаря має кілька важливих особливостей, що роблять його простим і зручним для використання. Однією з основних є його проста реалізація. Шифр можна легко застосувати вручну, просто замінюючи літери, або реалізувати за допомогою програмування. Ця особливість робить його ідеальним для навчання основ криптографії. Крім того, шифр має циклічність. Якщо ключ дорівнює 33 або є кратним кількості літер в алфавіті, текст залишається незмінним. Наприклад, для українського алфавіту зсув на 33 не змінює жодної літери, оскільки кожна літера

повертається на своє початкове місце. Ще однією важливою особливістю є фіксований зсув. Всі літери в тексті зміщуються на однакову кількість позицій, що робить шифр моноалфавітним. Через це кожен символ відкритого тексту завжди замінюється на один і той самий символ у шифрованому тексті, що, у свою чергу, знижує криптографічну стійкість шифру. Іншою особливістю є однаковий алфавіт для шифрування та розшифрування. Це означає, що всі літери або символи в алфавіті мають чітко визначену послідовність, що дозволяє застосовувати алгоритм до всього тексту без необхідності впроваджувати додаткові правила для різних символів.

До переваг шифру Цезаря можна віднести легкість використання, оскільки алгоритм є дуже простим, і для його реалізації не потрібні складні математичні обчислення або спеціалізовані інструменти. Це робить шифр доступним навіть для тих, хто тільки починає вивчати криптографію. Також шифр має мінімальні вимоги до ресурсів, що дозволяє його застосовувати навіть без комп'ютерної техніки – вручну на папері. Це важливо у випадках, коли доступ до технологій обмежений. Шифр Цезаря дуже швидкий у виконанні завдяки своїй простоті, що робить його ідеальним для шифрування коротких повідомлень.

Шифр Цезаря має кілька слабких сторін. Однією з основних є вразливість до частотного аналізу. Оскільки кожна буква відкритого тексту замінюється на одну й ту саму літеру, частота символів у зашифрованому тексті зберігається такою ж, як і в оригінальному тексті. Це дозволяє легко розшифрувати повідомлення, знаючи лише загальну частоту літер у мові. Наприклад, літера "О" в українському тексті зустрічається значно частіше за інші, тому її можна швидко ідентифікувати навіть без знання зсуву. Ще однією проблемою є малий розмір ключового простору. Оскільки можливі лише 33 різних зсувів (для українського алфавіту, оскільки зсув на 0 не змінює текст), шифр Цезаря можна легко зламати методом повного перебору, спробувавши всі можливі варіанти за кілька секунд. Крім того, шифр має недостатню складність. Моноалфавітність шифру, коли кожна літера замінюється на одну й ту саму, робить його дуже вразливим до сучасних методів криптоаналізу, зокрема до методів частотного аналізу. Це дозволяє швидко відновити оригінальний текст навіть без знання ключа. І, нарешті, шифр Цезаря має залежність від алфавіту. Він працює лише з текстами, що складаються з літер одного алфавіту.

Якщо в тексті є цифри, розділові знаки або текст на кількох мовах, застосування шифру ускладнюється, що обмежує його універсальність.

1.12.2 Шифр простої заміни

Шифр простої заміни є одним із базових видів моноалфавітних шифрів. Суть цього шифру полягає в тому, що кожен символ відкритого тексту замінюється на певний символ за певним правилом, яке визначається за допомогою ключа. Важливою особливістю є те, що кожен символ відкритого тексту замінюється на один і той самий символ шифрованого тексту, незалежно від його місця в тексті. Таким чином, застосовується постійне правило заміни для всіх літер або символів.

Процес шифрування за допомогою шифру простої заміни виглядає наступним чином:

1. Вибір ключа. Ключем шифру є набір замін для кожної букви алфавіту. Наприклад, для українського алфавіту, де є 33 літери, ключом може бути будь-яка перестановка цих літер. Це означає, що буква "А" може бути замінена на "О", "Б" – на "М", і так далі для всіх літер.

А→О	Е→Й	Ї→Ф	О→Ц	Ф→Ї	Ь→І
Б→М	Є→Р	Й→Г	П→Л	Х→Ж	Ю→Б
В→Х	Ж→У	К→Д	Р→Щ	Ц→С	Я→З
Г→Я	З→Ь	Л→К	С→Н	Ч→Є	
Ґ→Е	И→Ф	М→Ш	Т→И	Ш→Ч	
Д→А	І→Ю	Н→Т	У→В	Щ→Ґ	

Рисунок 1.3 – Алфавіт при шифрі простої заміни

2. Шифрування. Для кожної літери відкритого тексту шукається її відповідник у ключі та замінюється на відповідну букву шифрованого тексту. Наприклад, якщо в ключі зазначено, що "А" замінюється на "О", то кожну "А" в тексті буде замінено на "О". Це повторюється для всіх літер в тексті.
3. Розшифрування. Для того, щоб розшифрувати повідомлення, необхідно мати той самий ключ. Розшифрування полягає в заміні кожної літери за тим самим принципом, але на протилежну букву (тобто, якщо "А" заміняється на "О", то "О" потрібно замінити назад на "А").

Приклад шифрування. Візьмемо слово ЗНАННЯ, використовуючи таблицю заміни при шифруванні отримуємо «ЬТОТТЗ»

В шифрі простої заміни кожен символ відкритого тексту замінюється на конкретний символ шифрованого тексту. Відмінність від шифру Цезаря полягає в тому, що заміни можуть бути будь-якими (не лише зсувами), тому можливі всі види перестановок букв.

Відсутність циклічності. На відміну від шифру Цезаря, де можна визначити циклічність (наприклад, зсув на 33 символів повертає текст до початкового вигляду), в шифрі простої заміни немає такого ефекту. Кожен символ може бути замінений на інший символ незалежно від їхнього місця в алфавіті.

Використання ключа. Ключ у цьому шифрі є перестановкою всіх літер алфавіту. Це робить шифр більш складним для зламування порівняно з шифром Цезаря, оскільки кількість можливих перестановок для алфавіту з 33 літер складає 33!.

Шифр простої заміни має кілька сильних та слабких сторін, які визначають його ефективність у різних умовах. Однією з основних сильних сторін є підвищена криптографічна стійкість у порівнянні з шифром Цезаря. Оскільки кожна буква відкритого тексту може бути замінена на будь-яку іншу букву, шифр простої заміни має більший розмір ключового простору, що робить його більш стійким до атак методом перебору. Якщо шифр Цезаря можна зламати за допомогою 33 спроб (з урахуванням можливих зсувів), то для шифру простої заміни кількість варіантів для українського алфавіту становить 33!, що в рази більше і робить процес зламування значно складнішим.

Ще однією перевагою є простота реалізації. Шифр простої заміни можна застосувати як вручну, так і за допомогою комп’ютерних програм. Це дозволяє створювати ефективні інструменти для швидкого шифрування та розшифрування повідомлень. Крім того, шифр простої заміни не має обмежень на кількість зсувів. На відміну від шифру Цезаря, де літери зміщуються на фіксовану кількість позицій, шифр простої заміни дозволяє використовувати повну перестановку всіх літер алфавіту, що значно підвищує його складність і варіативність у виборі ключа.

Однак шифр простої заміни має і свої слабкі сторони. Основним недоліком є його вразливість до частотного аналізу. У будь-якій мові деякі літери зустрічаються частіше за інші (наприклад, "О", "І", "П"). Оскільки кожна буква

відкритого тексту замінюється на одну й ту саму літеру, частота символів у шифрованому тексті залишається такою ж, як і в відкритому.

Це дає змогу злому за допомогою частотного аналізу, коли криптоаналітик може співвіднести найбільш часто зустрічаються символи в шифрованому тексті з найпоширенішими літерами в мові. Іншим недоліком є малий розмір ключа. Хоча кількість можливих перестановок для українського алфавіту велика (33!), для малих алфавітів або обмежених наборів символів кількість можливих ключів значно зменшується, що робить шифр вразливим до простого методу перебору. І, зрештою, шифр простої заміни не відповідає сучасним стандартам безпеки. Хоча він є більш стійким, ніж шифр Цезаря, сучасні методи криптоаналізу, зокрема частотний аналіз, дозволяють легко зламати цей шифр, якщо він застосовується до великих обсягів тексту, що робить його неефективним для сучасного використання в умовах високої вимоги до безпеки.

1.12.3 Шифр Атбаш

Шифр Атбаш [7] є класичним методом заміни в криптографії, який використовується для перетворення відкритого тексту в зашифрований. Це моноалфавітний шифр, що працює на принципі заміни кожної літери алфавіту на її симетричну пару. Заміна здійснюється таким чином: перша літера алфавіту (А) замінюється на останню (Я), друга літера (Б) — на передостанню (Ю), третя (В) — на третю з кінця (Ь), і так далі. Для українського алфавіту цей процес виглядає наступним чином:

А→Я	Е→Ц	Ї→Р	О→К	Ф→Ж	Ь→В
Б→Ю	Є→Х	Й→П	П→Й	Х→Є	Ю→Б
В→Ь	Ж→Ф	К→О	Р→Ї	Ц→Е	Я→А
Г→Щ	З→У	Л→Н	С→І	Ч→Д	
Ґ→Ш	И→Т	М→М	Т→И	Ш→Ґ	
Д→Ч	І→С	Н→Л	У→З	Щ→Г	

Рисунок 1.4 – Алфавіт при шифрі Атбаш

Така операція створює симетричну структуру, у якій кожна буква має чітко визначену пару. Алгоритм шифру Атбаш простий і легкий у реалізації. У разі

необхідності зворотна операція (розшифрування) виконується тим самим способом, оскільки схема заміни симетрична.

Наприклад слово ПІДРУЧНИК буде зашифровано в «ЙСЧЇЗДЛТО»

Алгоритм шифру є дуже простим і не вимагає складних математичних операцій. Для шифрування та розшифрування застосовується одна і та сама операція заміни, що робить його симетричним: щоб розшифрувати текст, достатньо застосувати той самий метод, що і для шифрування. Ця особливість надає шифру Атбаш певну зручність, оскільки той самий ключ можна використовувати для обох процесів. Однак, шифр має певні обмеження, зокрема він працює лише з алфавітними символами. Тому для текстів, що містять цифри, розділові знаки або пробіли, цей шифр не підходить.

Шифр Атбаш є простим і швидкісним для виконання. Завдяки чітко визначеній заміні букв, алгоритм можна застосувати швидко, навіть вручну. Він не потребує великих обчислювальних ресурсів, що робить його зручним у ситуаціях, коли ресурси обмежені. Цей шифр також має історичне значення, оскільки є одним з найстаріших відомих шифрів, що використовувався ще в Стародавньому Ізраїлі для військових і дипломатичних повідомлень.

Проте шифр Атбаш має кілька суттєвих недоліків. Оскільки він є моноалфавітним, кожна літера в шифрованому тексті завжди замінюється на одну й ту ж саму літеру, що залишає відкритими закономірності частоти символів. Це робить його вразливим до частотного аналізу: вивчаючи частоту літер в шифрованому тексті, можна легко відновити відкритий текст, оскільки певні літери в мові з'являються значно частіше за інші і її можна ідентифікувати навіть без розуміння зсуву, застосованого в шифрі. Крім того, кількість можливих перестановок у шифрі Атбаш обмежена кількістю літер в алфавіті, що робить його дуже вразливим до атак методом перебору, коли криптоаналітик може перевірити всі можливі варіанти замін. Це означає, що шифр Атбаш має обмежену криптографічну стійкість і не підходить для захисту великих обсягів даних чи для використання в умовах, де потрібно забезпечити високий рівень безпеки. Відсутність складності в алгоритмі також обмежує його застосування в сучасних системах комунікації, де необхідно мати більш стійкі методи шифрування для захисту конфіденційної інформації.

1.12.4 Шифр Плейфера

Алгоритм шифру Плейфейра [8] передбачає два основні етапи: створення таблиці і сам процес шифрування. Для початку необхідно створити таблицю 6х6, використовуючи певне ключове слово або фразу. Алгоритм роботи такий:

Спочатку необхідно вибрати ключове слово, яке використовуватиметься для заповнення таблиці. Усі букви в цьому слові додаються до таблиці без повторів, а потім заповнюються решта місць таблиці, використовуючи решту літер алфавіту. Важливо зауважити, що в таблиці буде лише 33 літери, а пусті комірки будуть заповнені розділовими знаками.

Наприклад, якщо ключове слово — "ЕЛЕКТРОНІКА", таблиця виглядатиме так:

Е	Л	К	Т	Р	О
Н	І	А	Б	В	Г
Ґ	Д	Є	Ж	З	И
Ї	Й	М	П	С	У
Ф	Х	Ц	Ч	Ш	Щ
Ь	Ю	Я	.	,	!

Рисунок 1.5 –Алфавіту при шифру Плейфейра

Для шифрування ми візьмемо наступну фразу «Вчитися завжди добре», її необхідно розбити на пари літер «ВЧ ИТ ИС ЯЗ АВ ЖД ИД ОБ РЕ» і вже використовуючи таблицю зашифрувати за наступними правилами:

Якщо обидві літери пари знаходяться в одному рядку, кожна літера замінюється на ту, що знаходиться праворуч (якщо літеру на правому кінці, вона замінюється на першу літеру того ж рядка).

Якщо обидві літери пари знаходяться в одному стовпці, кожна літера замінюється на ту, що знаходиться нижче (якщо літера внизу, вона замінюється на першу літеру того ж стовпця).

Якщо літери пари знаходяться в різних рядках і стовпцях, кожна літера замінюється на літеру, що знаходиться на перехресті того ж рядка, де знаходиться перша літера пари, і того стовпця, де знаходиться друга літера [1].

Після шифрування фрази ми отримали наступне:

«БШ ЖО ЗУ ,Є БГ ЗЄ ҐЄ ТГ ОЛ»

Розшифрування відбувається за допомогою зворотних правил заміни.

1.13 Порівняння шифрів

Моноалфавітні шифри, такі як шифр Цезаря, шифр простої заміни, шифр Атбаш і шифр Плейфейра, мають схожі принципи роботи, але значно відрізняються за складністю та рівнем безпеки. Кожен із них має свої сильні та слабкі сторони, що робить їх більше чи менше придатними для різних умов використання. Порівняємо ці шифри між собою, зокрема з точки зору криптографічної стійкості, простоти реалізації та можливостей застосування.

Простота реалізації та використання:

Шифр Цезаря є найпростішим і найшвидшим для реалізації. Це класичний шифр, який можна виконати вручну без будь-яких спеціальних інструментів. Для нього достатньо лише вибрати зсув, який змінює літери відкритого тексту на певну кількість позицій у алфавіті.

Шифр Атбаш також дуже простий і інтуїтивно зрозумілий. Тут не потрібно вибирати зсув або перестановку, оскільки літери замінюються на свої симетричні пари. Для цього шифру достатньо просто знати алфавіт і його зворотний порядок.

Шифр простої заміни потребує більше зусиль, оскільки для його реалізації потрібно створити перестановку всіх літер алфавіту, що робить алгоритм складнішим порівняно з шифром Цезаря чи Атбаш. Однак сам процес шифрування чи розшифрування простий: просто замінюємо одну букву на іншу за заздалегідь визначеним правилом.

Шифр Плейфейра є найбільш складним серед розглянутих шифрів. Для його реалізації потрібна таблиця, в якій розміщені букви алфавіту з використанням ключового слова. Це вимагає більше зусиль при підготовці та розробці алгоритму, але він має більшу криптографічну стійкість.

Безпека і криптографічна стійкість:

Шифр Цезаря має низький рівень безпеки. Оскільки існує лише 33 можливих варіантів зсуву (для українського алфавіту), шифр легко зламати методом перебору або частотного аналізу. Цей шифр можна швидко зламати навіть за допомогою простих інструментів або вручну. Тому його застосовують лише в умовах, де безпека не є критично важливою.

Шифр Атбаш має схожу проблему з шифром Цезаря: він є моноалфавітним, тобто кожна буква відкритого тексту замінюється на одну і ту ж букву в зашифрованому тексті. Завдяки цьому частотний аналіз також може легко зламати цей шифр, особливо для великих текстів, де закономірності частоти букв сильно проявляються.

Шифр простої заміни має більшу криптографічну стійкість порівняно з шифром Цезаря та Атбаш. В ньому використовується повна перестановка всіх літер алфавіту, що дає величезну кількість варіантів шифрування (33! для українського алфавіту). Однак цей шифр також вразливий до частотного аналізу, хоча і в меншій мірі, оскільки частотні закономірності можуть бути збережені в шифрованому тексті.

Шифр Плейфейра має найбільшу стійкість до криптоаналізу серед моноалфавітних шифрів. Оскільки він використовує заміну пар літер, частотний аналіз стає набагато складнішим. Однак шифр все ж має вразливість: при використанні стандартних таблиць Плейфейра, він може бути зламаний, але для цього знадобиться більше часу та зусиль, ніж для шифрів Цезаря чи Атбаш.

1.14 Висновки про використання шифрів

Шифр Цезаря підходить для використання в ситуаціях, коли необхідна швидка і проста криптографія, але без високих вимог до безпеки. Це ідеальний варіант для навчальних цілей або для використання в ситуаціях, де обсяг зашифрованої інформації дуже малий і криптографічна безпека не є пріоритетом.

Шифр Атбаш має обмежене використання і більше підходить для демонстраційних цілей або розваг, оскільки його стійкість до аналізу дуже низька.

Шифр простої заміни є кращим варіантом серед моноалфавітних шифрів, коли потрібно забезпечити певний рівень захисту, але також він не підходить для серйозних криптографічних завдань через свою вразливість до частотного аналізу.

Шифр Плейфейра є найбільш стійким серед цих шифрів для використання у випадках, де не потрібно дуже високої криптографічної безпеки, але важлива стійкість до частотного аналізу. Він може бути корисний для захисту менш критичної інформації, де вищі вимоги до захисту неможливі через обмеження ресурсів.

Моноалфавітні шифри можна використовувати коли є обмеження ресурсів (наприклад, низька обчислювальна потужність або обмежена пам'ять) моноалфавітні шифри можуть бути корисними для швидкого та простого шифрування. Оскільки вони не потребують великих обчислювальних потужностей, їх можна реалізувати на простих пристроях або навіть вручну.

Моноалфавітні алгоритми шифрування, незважаючи на їхню простоту, залишаються важливим об'єктом для дослідження в сучасній криптографії. Вивчення таких алгоритмів дозволяє глибше розуміти основи шифрування. Хоча моноалфавітні шифри поступаються сучасним методам за рівнем криптостійкості, вони досі знаходять своє застосування у певних галузях, що вимагає подальшого аналізу.

Одним напрямом дослідження є оцінка вразливості моноалфавітних алгоритмів у порівнянні з більш складними методами. Аналіз слабких місць цих шифрів, таких як вразливість до частотного аналізу, дозволяє краще розуміти способи їхнього зламу. Це, у свою чергу, сприяє вдосконаленню сучасних систем захисту, які інтегрують елементи простих шифрувальних технік у більш складні архітектури.

Моноалфавітні алгоритми можуть бути корисними у випадках, коли необхідна базова форма захисту або коли конфіденційність даних не є критично важливою. Наприклад, у сфері освіти, гейміфікації або при створенні низькобюджетних рішень для малих організацій, які не мають доступу до складного програмного забезпечення. Таким чином, дослідження цих алгоритмів також допомагають розробляти прості рішення для таких сценаріїв.

Отже, проведення досліджень у цій галузі має значний практичний і теоретичний потенціал. Моноалфавітні алгоритми залишаються важливими не лише як частина історії криптографії, але й як база для створення нових методів захисту інформації. Це забезпечує актуальність та необхідність подальшого вивчення цих методів у сучасному світі.

1.15 Постановка завдання

Метою роботи є розробка електронної системи захисту інформації на базі моноалфавітного алгоритму шифрування, яка повинна забезпечити необхідний

рівень криптостійкості при шифруванні вхідних повідомлень шифром Цезаря з ключовим словом.

Для розроблення електронної системи захисту інформації на базі моноалфавітного алгоритму шифрування необхідно виконати наступне:

1. Визначити завдання та основні функції, які буде виконувати системи захисту інформації.

2. Виконати аналіз оцінки криптостійкості моноалфавітних шифрів та способів забезпечення необхідного рівня криптостійкості при шифруванні шифром Цезаря з ключовим словом. Спираючись на результати досліджень уточнити алгоритм роботи системи формування моноалфавітних шифрів.

3. Розробити схему алгоритму роботи та схему електричну структурну системи захисту інформації на базі моноалфавітного алгоритму шифрування.

4. Розробити схему електричну функціональну та схему електричну принципову системи, що розроблюється.

5. Розрахувати собівартість та ціну системи.

2 НАУКОВО-ДОСЛІДНА ЧАСТИНА

Оцінка стійкості шифруючих перетворень моноалфавітної заміни з використанням генетичного алгоритму

Основою більшості систем інформаційної безпеки є ефективне використання криптографічних методів і алгоритмів. В даний час існує велика кількість різних алгоритмів шифрування, таких як ДСТУ ГОСТ 28147-2009, DES, IDEA.

Як відомо, злам будь-якого шифру – це питання часу і коштів. Однак різні шифри мають різний рівень стійкості до злому. Для його оцінки використовуються відомі методи криптоаналізу, засновані на таких атаках, як атака повним перебором, за відкритим текстом, за шаблоном, за словник, за частотою символів у тексті, а також різні евристичні атаки. Крім того, знання історії атак і вразливих місць у реалізації алгоритмів захисту, а також розумінні причин виникнення атак, являється одним із необхідних умов розробки захищених криптосистем.

Наразі всі сучасні системи криптографічного захисту інформації ґрунтуються на принципах симетричного і асиметричного шифрування. У симетричних криптосистемах шифрування та дешифрування інформації здійснюється за допомогою одного секретного ключа К, розсекречення якого веде до компрометації всієї системи.

Здатність криптосистеми протистояти атакам криптоаналітика називається стійкістю, яка може визначатися інтервалом часу необхідним для розкриття шифру. Стійкою вважається криптосистема, яка для успішної атаки вимагає від противника недосяжних обчислювальних ресурсів, недосяжного обсягу перехоплених відкритих і зашифрованих повідомлень або такого розкриття, що після його закінчення захищена інформація буде вже не актуальна.

Найбільшу криптостійкість мають сучасні симетричні криптосистеми, такі як DES, IDEA, ГОСТ. Криптостійкість класичних симетричних шифрів істотно нижча, тому вони є найбільш привабливими для криптоаналітика при оцінці ефективності методів та алгоритмів криптоаналізу. Крім того, криптостійкість шифру безпосередньо залежить від типу атаки. Для успішного дешифрування потрібно зробити вибір ефективного методу криптоаналізу для конкретного типу криптосистем.

Реалізацію криптоатаки можна розглядати як задачу пошуку секретного ключа в просторі ключів шифрування. Якщо при цьому ввести деякий мінімізуючий функціонал, то цю задачу можна розглядати як задачу оптимізації. У випадку реалізації атаки на симетричну криптосистему таким функціоналом може бути математичний вираз, який показує відстань між отриманим відкритим текстом на поточній стадії атаки та початковим відкритим текстом, який необхідно отримати. При цьому метрика такої відстані може бути будь-якою, головне, щоб вона дозволяла визначати схожість відкритих текстів. У цьому випадку близькість поточного та реального секретного ключа шифрування буде призводити до схожості відкритих текстів. Таким чином, маючи мінімізуючий функціонал і систему обмежень, що накладаються на параметри ключа шифрування та відкритого тексту, задача криптоаналізу зводиться до розв'язання оптимізаційної задачі.

Для вирішення задач оптимізації існує велика кількість традиційних алгоритмів. Їх загальним недоліком є те, що при збільшенні розмірності простору пошуку кількість точок даного простору значно збільшується, що призводить до необхідності залучення великих обчислювальних та часових ресурсів. Однак, у вирішенні задачі криптоаналізу основною метою є пошук не оптимального рішення, а близького до оптимального. З цих позицій ефективним підходом до вирішення цього завдання є використання генетичного алгоритму як методу випадкового спрямованого пошуку розв'язання оптимізаційних завдань. Розглянемо такий підхід до реалізації криптоаналітичної атаки з метою оцінки стійкості шифру моноалфавітної заміни.

Як приклад розглянемо шифр моноалфавітної заміни, у якому довжина ключа шифрування дорівнює потужності алфавіту. Для наочної демонстрації роботи даного шифру достатньо виписати під заданим алфавітом той самий алфавіт, але у випадковому порядку його символів або, наприклад, зі зміщенням. Записаний у такий спосіб алфавіт називають алфавітом заміни.

Нехай A= Ψ = {A,B … X,Y,Z} – вихідний алфавіт, що складається з безлічі великих англійських букв (потужність дорівнює 26), K = (a_1 a_2 … a_{26}) – ключ шифрування, де ∀ i ∈ [1 … 26]: a_i ∈ Ψ і ∀ a_i і $a_k \to a_i \neq a_k$. Остання умова визначає факт неповторюваності символів алфавіту в ключі шифрування. Крім того, над символами відкритого тексту, які не входять до алфавіту, шифрування не проводиться. Це можна формалізувати так: x ∈ Ψ, то F_k (x) = x.

Існує також зворотна функція $H_k=F_k^{-1}$, що дозволяє однозначно реалізувати процедуру дешифрування закритого тексту.

На рисунку 2.1 наведена схема шифрування та дешифрування для алгоритму.

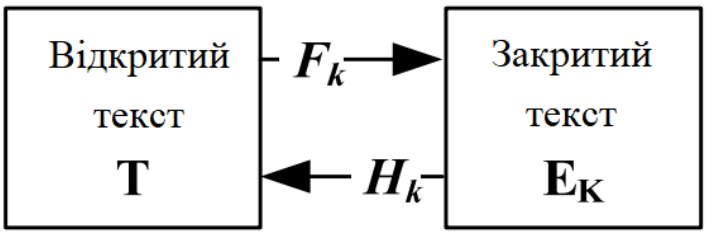


Рисунок 2.1 – Схема симетричного шифрування

Тут $T= (t1)_{1\le i\le N}$ – відкритий текст, N – довжина відкритого тексту.

$E_k= (e_i)_{1\le i\le N} = F_k(T)\ \forall\ e_i \in E_k, e_i = F_k(t_i)$, де $t_i \in T$.

У реальних завданнях криптоаналізу аналітику відомий лише зашифрований текст E, тоді як секретний ключ K та відкритий текст T невідомі. Завдання криптоаналізу полягає у виявленні цього секретного ключа, щоб на основі відомого алгоритму шифрування отримати вихідний або близький до нього відкритий текст T.

Зазначимо, що шифри моноалфавітної заміни легко розкриваються за допомогою методу частотного аналізу, оскільки не змінюють частоти використання символів у повідомленні. Однак для реалізації такого роду атаки необхідно мати досить довгі шифротексти, що не завжди задовольняє реальні умови. З використанням методу повного перебору всіх можливих варіантів перестановок ключа (методу грубої сили) потрібно залучення великих обчислювальних і часових ресурсів, оскільки потужність ключового простору становитиме величину, що дорівнює $26! \approx 288{,}4$, що не дозволяє застосувати цей вид атаки для ключів великої розмірності.

Для оцінки стійкості розглянутого шифру моноалфавітної заміни розроблено генетичний алгоритм, що складається з наступних етапів:

1) Створення початкової популяції.
2) Визначення функцій пристосованості для особин популяції (оцінка їхньої якості).
3) Початок циклу:

– селекція двох батьківських хромосом з поточної популяції;
– схрещування та мутація хромосом;
– обчислення функцій пристосованості для всіх хромосом;
– редукція найгірших хромосом та формування нового покоління хромосом;
– якщо виконується умова зупинки, то перехід у кінець циклу, інакше повернення на початок;

4) Кінець циклу.

Розглянутий алгоритм можна подати у вигляді блок-схеми, наведеної на рисунку 2.2.

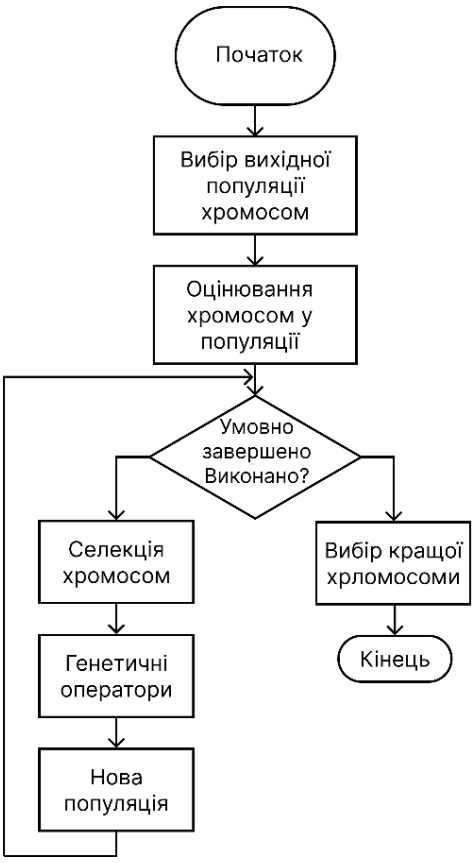


Рисунок 2.2 – Блок схема генетичного алгоритму

У розробленому генетичному алгоритмі хромосома є рішенням завдання (ключ шифрування). В даному випадку, маючи вихідний алфавіт Ψ = (A,B, … X,Y,Z) отримаємо:

P = ([a_1 a_2 … a_{26}] generation, fitness), де ∀ i ∈ [1...26]: a_i ∈ Ψ та ∀ a_i і a_k:

$i \neq k \rightarrow a_i \neq a_k$, generation - номер покоління хромосоми, fitness – речове число, що дорівнює значенню функції пристосованості хромосоми в популяції.

Поточні хромосомні набори даних зберігаються у спеціальному масиві, яким є генофонд. Для створення початкової популяції відбувається її ініціалізація двома випадковими кандидатами (хромосомами), які генеруватимуть наступне потомство. Число кандидатів у генофонді завжди буде парним.

Розглянемо дві батьківські хромосоми ($X=[a_1 \dots a_{26}]$) та ($Y= [\delta_1 \dots \delta_{26}]$). Для визначення їх точки схрещування випадково вибирається число $n \in [1...25]$. В результаті роботи оператора схрещування від батьківських хромосом X та Y будуть отримані дві дочірні хромосоми U та V:

$$U = [a_1 \dots a_n \delta_{n+1} \dots \delta_{26}],$$

$$V = [\delta_1 \dots \delta_n a_{n+1} \dots a_{26}],$$

В результаті схрещування завжди виходить парне число нових хромосом.

Процес відбору хромосом можна виконувати двома способами. Перший спосіб передбачає вибір кращих хромосом $k=p*n$, де n – поточний розмір генофонду, p – ймовірність схрещування ($0<p\leq1$). При цьому схрещування здійснюється між кожними двома послідовними елементами. Другий спосіб передбачає вибір перших кандидатів (кращі елементи), а також останніх елементів. Схрещування здійснюється між цими елементами.

У поточній реалізації генетичного алгоритму використовується другий спосіб відбору хромосом, щоб «погані» батьки мали можливість брати участь у процесі схрещування.

В результаті роботи оператора схрещування можуть бути отримані дефектні хромосоми, у яких позначення різних алелей збігаються тобто $\exists\ i\in[1,n]$ і $k\in[n+1,26]$ такі, що $a_i = \delta_i$.

Якщо $\Psi = \{A,B,C,D,E,F,G\}$ – вихідний алфавіт, а X = [AECBGFD] та Y = [DBCFAGE] - батьківські хромосоми та обрана точка схрещування n=3, то будуть отримані дочірні хромосоми U=[AECFAGE] та V=[DBCBGFD].

Розглянемо процес виправлення дефектних хромосом (рис. 2.3).

Як видно з рисунку, кожна з дочірніх хромосом має дві однакові алелі. У процесі виправлення хромосом шляхом заміни надлишкових алелів недостатнім значенням отримаємо нові дочірні хромосоми. U = [AECFBGD] та V = [DBCAGFE].

	1	2	n=3	4	5	6	7
X	A	E	C	B	G	F	D
Y	D	B	C	F	A	G	E
U	A	E	C	F	A	G	E
V	D	B	C	B	G	F	D
U'	A	E	C	F	B	G	D
V'	D	B	C	A	G	F	E

Рисунок 2.3 - Виправлення дефектних хромосом

Для покращення збіжності генетичного алгоритму застосовується мутація хромосом. Розглянемо хромосому X = [a_1 … a_{26}]. Для реалізації оператора мутації випадково вибираються два цілі числа n∈ [1...26] і m∈ [1...26], n≠m. Потім створюється нова хромосома Y шляхом дублювання хромосоми X та заміни місцями позиції елементів a_m та a_n. В результаті мутації створюється хромосома, здатна в майбутньому дати найкраще потомство.

Розглянемо функцію пристосованості хромосом. Припустимо, що E є зашифрованим текстом і K – поточний ключ шифрування. Для отримання відкритого тексту U цей ключ необхідно використовувати для розшифрування зашифрованого тексту E за наступною формулою:

$$U = f_K^{-1}(E) \tag{2.1}$$

У цьому випадку значення пристосованості рішення До повинно відображати ступінь коректності відкритого тексту U з урахуванням того, що оригінальний відкритий текст T залишається невідомим. Іншими словами, це значення має визначити, наскільки відкритий текст U є логічно прийнятним текстом (його читабельність)

Щоб визначити ступінь читабельності відкритого тексту U необхідно провести його глибокий семантичний і синтаксичний аналіз. Однак це завдання виходить за рамки дослідження. З іншого боку, ступінь його читабельності можна

оцінити шляхом підрахунку кількості невідомих слів, які у цьому тексті. Це досягається шляхом використання наступної формули:

$$Fitness(K)=100*A/B \quad (2.2)$$

де B – кількість всіх слів отриманому відкритому тексті U; A – кількість невідомих слів у тексті (що не містяться у словнику), причому $0 \neq A \neq B \rightarrow 0 \neq Fitness(K) \neq 100$.

Найкращим рішенням є те, що має нульове значення функції пристосованості або принаймні мінімальне значення.

Таким чином, для практичної оцінки пристосованості хромосом у популяції необхідний деякий словник, який дозволить виявляти некоректні слова в відкритому тексті U, що отримується. Під некоректними в даному випадку слід розуміти слова, не включені в словник.

Слід зазначити, що існує проблема реалізації криптоатаки на тексти малої довжини. Припустимо, що зашифрований текст E є текстом малої довжини, наприклад, E=[EONMVNKO] та нехай на різних циклах роботи генетичного алгоритму знайдено чотири ключі, в результаті застосування яких на етапі розшифрування повідомлення E отримаємо такі псевдооригінальні відкриті тексти (табл. 2.1).

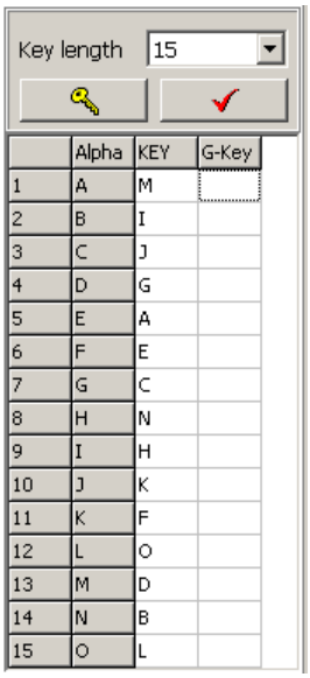
Таблиця 2.1 – Результат реалізації криптоатаки на тексти малої величини

Ключ	Відкритий текст	Значення функції пристосованості	Переклад
K1	HE IS NICE	0	Він красивий
K2	AS IN PIGS	0	Як у свині
K3	HE IS TIME	0	Він є час
K4	BE AT CAKE	0	Бути в торті

Цей приклад показує, що можна знайти різні псевдооригінальні відкриті тексти, але при цьому залишається проблема вибору з них близько до

оригінального тексту. Очевидно, що для текстів великої довжини дана проблема є набагато менш актуальною, так як при цьому різко зменшується ймовірність отримання псевдооригінальних відкритих текстів.

Розглянемо приклад роботи генетичного алгоритму. Згенеруємо ключ шифрування довжиною 15 символів (рис. 4).



	Alpha	KEY	G-Key
1	A	M	
2	B	I	
3	C	J	
4	D	G	
5	E	A	
6	F	E	
7	G	C	
8	H	N	
9	I	H	
10	J	K	
11	K	F	
12	L	O	
13	M	D	
14	N	B	
15	O	L	

Рисунок – 2.4 Ключ шифрування відкритого тексту

Розглянемо вихідний відкритий текст:

IF A BROKEN ARCHIVE DOES NOT CONTAIN A RECOVERY RECORD OR IF THE ARCHIVE IS NOT COMPLETELY RECOVERED DUE TO MAJOR DAMAGE, A SECOND STAGE IS PERFORMED. DURING THIS STAGE ONLY THE ARCHIVE STRUCTURE IS RECONSTRUCTED AND IT IS IMPOSSIBLE TO RECOVER FILES WHICH FAIL THE CRC VALIDATION, IT IS STILL POSSIBLE, HOWEVER, TO RECOVER UNDAMAGED FILES, WHICH WERE INACCESSIBLE DUE TO THE BROKEN ARCHIVE STRUCTURE. MOSTLY THIS IS USEFUL FOR NON-SOLID ARCHIVES. WHEN THE SECOND STAGE IS COMPLETED, THE RECONSTRUCTED ARCHIVE WILL BE SAVED.

У результаті шифрування відкритого тексту отримуємо наступний зашифрований текст:

HE M IRLFAB MRJNHVA GLAS BLT JLBTMHB M RAJLVARY RAJLRG LR HE TNA MRJNHVA HS BLT JLDPOATAOY RAJLVARAG GUA TL DMKLR GMDMCA, M SAJLBG STMCA HS PARELRDAG. GURHBC TNHS STMCA LBOY TNA MRJNHVA STRUJTURA HS RAJLBSTRUJTAG MBG HT HS HDPLSSHIOA TL RAJLVAR EHOAS WNHJN EMHO TNA JRJ VMOHGMTHLB, HT HS STHOO PLSSHIOA, NLWAVAR, TL RAJLVAR UBGMDMCAG EHOAS, WNHJN WARA HBMJJASSHIOA GUA TL TNA IRLFAB MRJNHVA STRUJTURA. DLSTOY TNHS HS USAEUO ELR BLB-SLOHG MRJNHVAS. WNAB TNA SAJLBG STMCA HS JLDPOATAG, TNA RAJLBSTRUJTAG MRJNHVA WHOO IA SMVAG.

Для реалізації криптоаналітичної атаки запустимо генетичний алгоритм, фітнес-функція якого представлена на рисунку 2.5.

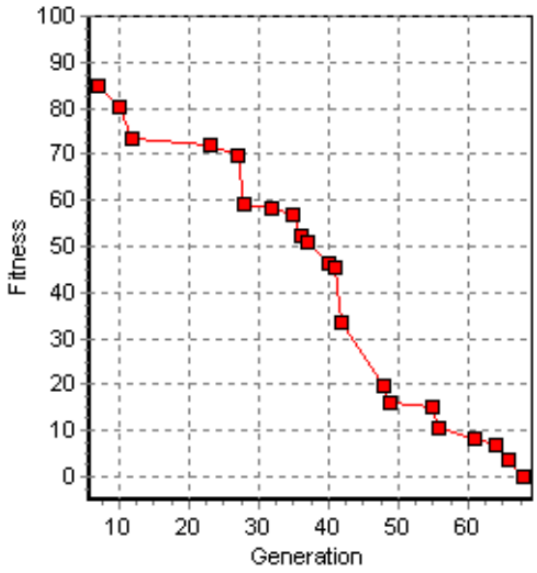


Рисунок 2.5 – Фітнес група генетичного алгоритму

У цьому випадку в результаті реалізації криптоаналітичної атаки з використанням генетичного алгоритму відновлено повністю відкритий текст і, відповідно, ключ шифрування. Процес розшифровки тексту займав 8,8 секунди, число поколінь склало 64, число популяцій – 6756. При збільшенні довжини шифротексту і ключа шифрування збільшується час роботи генетичного алгоритму, але це майже не впливає на кінцевий результат – на виході виходить читабельний відкритий текст, що підтверджує ефективність генетичного алгоритму.

У реалізації розробленої системи користувач може застосовувати або не застосовувати частотний аналіз тексту. Оцінимо, як змінюється ефективність алгоритму залежно від застосування частотного аналізу. Для цього було взято та зашифровано художній текст англійською мовою, що складається з 1179 слів. Усі параметри, крім використання або невикористання частотного аналізу залишалися незмінними під час дослідження. Такий розмір тексту обумовлений тим, що більше обсяг, тим ближче розподіл частот у тексті до стандартного розподілу.

У таблиці 2.2 представлені результати роботи генетичного алгоритму з використанням та без використання частотного аналізу тексту.

Таблиця 2.2 – Результати роботи генетичного алгоритму

№	Число поколінь		Час роботи, с	
	Без частотного аналізу	З частотним аналізом	Без частотного аналізу	З частотним аналізом
1	178	161	380,41	318,61
2	189	152	412,16	301,41
3	186	166	407,63	327,73
4	173	149	369,14	294,72
5	181	158	392,97	311,27
Середні значення параметрів				
	181	157	392,46	310,74

На рисунку 2.6 показані графіки функції пристосованості fitnes для першого циклу роботи генетичного алгоритму.

При використанні частотного аналізу можна значно скоротити час роботи генетичного алгоритму, так і кількість поколінь, необхідних для знаходження ключа шифрування. Це відбувається через те, що в першому поколінні є хромосома з низьким значенням функції пристосованості. Час, який програма витрачає обробку кожного покоління, не змінюється, але значно скорочується кількість поколінь.

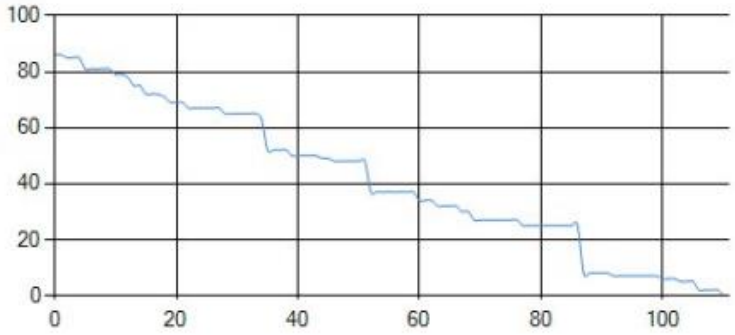


Рисунок 2.6 – Графік функцій для першого циклу роботи генетичного алгоритму без частотного аналізу

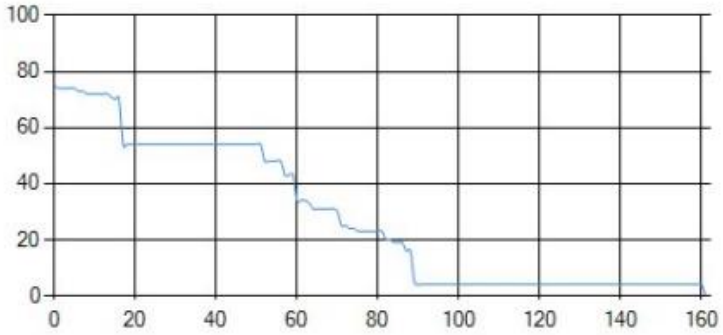


Рисунок 2.7 - Графік функцій для першого циклу роботи генетичного алгоритму з частотним аналізом

Слід зазначити, що ефективність частотного аналізу залежить від частотного розподілу символів відкритого тексту та відповідності цього стандартного розподілу для англійської мови. Таким чином, чим більший обсяг тексту аналізуватиметься, тим більшою буде різниця в ефективності роботи розробленої системи при використанні частотного аналізу і без нього.

Таким чином, генетичний алгоритм є ефективним інструментом оцінки стійкості шифруючих перетворень моноалфавітної заміни. Такі шифри не можна використовувати для шифрування в чистому вигляді, оскільки їхня криптостійкість є дуже низькою. Крім того, використання частотного аналізу дозволяє значно скоротити час роботи генетичного алгоритму, що підвищує ефективність його роботи та криптоаналітичної атаки загалом.

За результатами проведених досліджень доцільно використання генетичного алгоритму для оцінки криптостійкості зашифрованих повідомлень із застосуванням відповідних параметрів шифрування, таких як обраний ключ та ключове слово для моноалфавітної підстановки (шифр Цезаря з ключовим словом). Порівняння заданого рівня криптостійкості із розрахунковою оцінкою криптостійкості надає можливість підвищення фактичного рівня криптостійкості шляхом зміни параметрів шифрування на відповідній частині повідомлення із наступною зміною параметрів шифрування на наступній частині повідомлення. Така адаптивність дозволяє усунути частотні особливості зашифрованого тексту та за рахунок необхідної кількості циклів змін параметрів шифрування досягнути необхідного рівня криптостійкості та підвищити обсяг часу для дешифрування до неприпустимого для криптоаналітика рівня.

3 РОЗРОБКА АЛГОРИТМУ РОБОТИ ТА СХЕМИ ЕЛЕКТРИЧНОЇ СТРУКТУРНОЇ

3.1 Розробка алгоритму роботи системи

Для досягнення необхідного рівня криптостійкості при використанні моноалфавітної підстановки (шифр Цезаря з ключовим словом) потрібно знати вимоги передавача інформації про необхідний рівень її криптостійкості. За допомогою програмного забезпечення, що використовує генетичні алгоритми, необхідно здійснювати оцінку криптостійкості зашифрованих частин повідомлення із визначенням довжини повідомлення та відповідних параметрів шифрування. Робота системи шифрування повинна відбуватись відповідно до розробленого алгоритму, що наведений на рисунку 3.1.

1 блок: Введення вхідного повідомлення, що підлягає шифруванню.

2 блок: Аналіз вимог до криптостійкості повідомлення, що підлягає передачі по незахищеним каналам зв’язку.

3 блок: Формування параметрів шифрування для моноалфавітної підстановки (шифр Цезаря с ключовим словом).

4 блок: Визначення необхідної кількості циклів змін параметрів шифрування та кількості частин повідомлення для яких планується зміна параметрів шифрування.

5 блок: Визначення кількості циклів змін параметрів шифрування для відповідної кількості частин вхідного повідомлення. Якщо кількість циклів не досягнуто, то перехід на блок алгоритму 6 для початку шифрування.

6 блок: Початок формування таблиці Цезаря.

7 блок: Вибираємо український алфавіт.

8 блок: Вибираємо англійський алфавіт.

9 блок: Вказуємо інший алфавіт, якого немає в списку.

10 блок: Вводимо ключ К - це число, яке визначає зсув символів алфавіту вправо під час шифрування. Наприклад, для українського алфавіту межі К складають від 0 до 33.

11 блок: Виконуємо зсув символів алфавіту на К символів. Кожен символ зміщується починаючи з нової позиції. Наприклад, якщо К – 5, то алфавіт буде починатися з літери «Д».

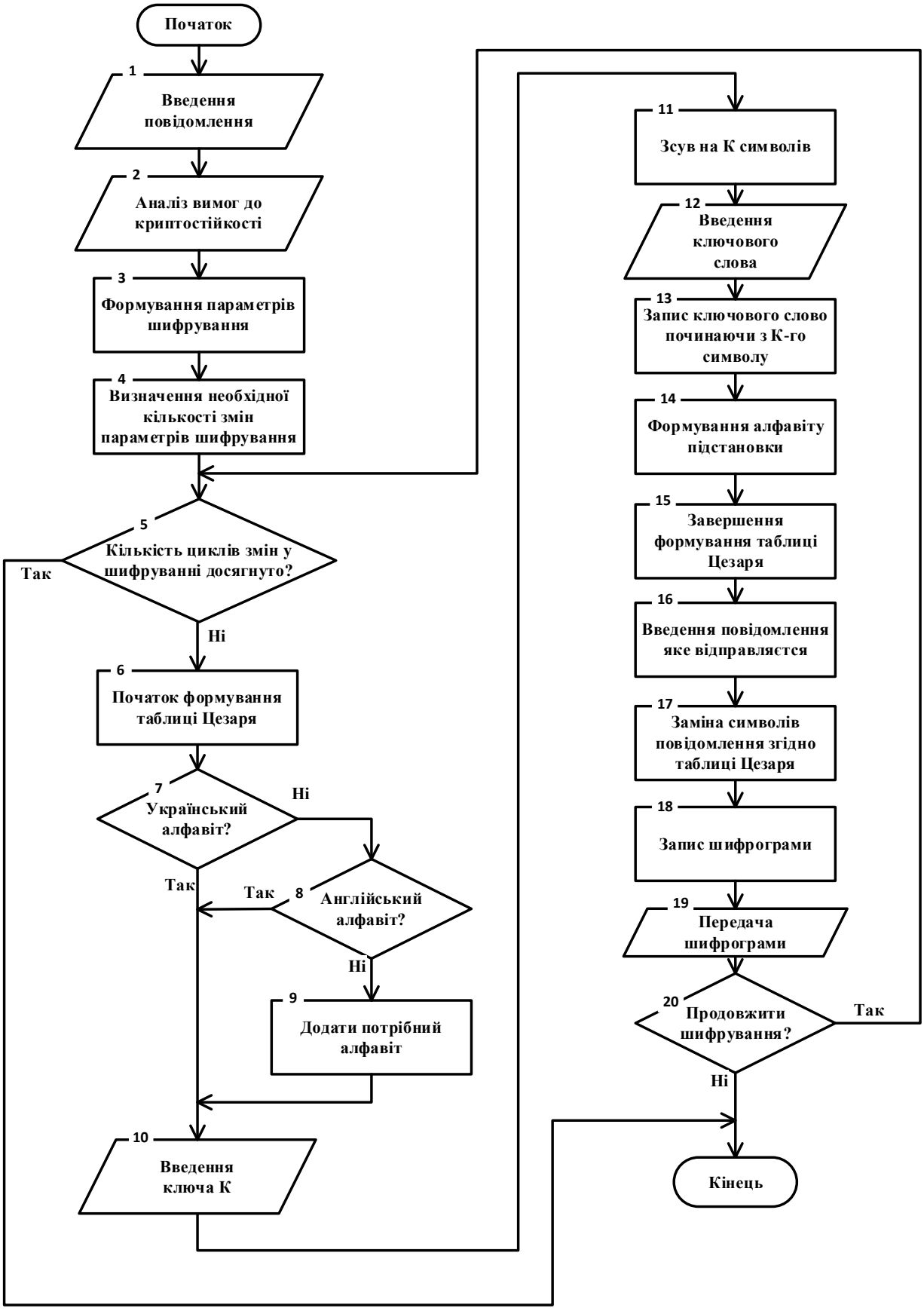


Рисунок 3.1 – Схема алгоритму

12 блок: Вводимо ключове слово, яке задає користувач. Його переводимо на числовий ключ: кожна літера отримує числове значення залежно від місця в алфавітному порядку (Наприклад, для українського алфавіту: А=0, Б=1, … , Я=33). Це значення використовується для зсуву символів під час шифрування чи розшифрування.

13 блок: Запис ключового слова, яке буде починатися з 8-ї позиції блоку. Наприклад, якщо ключове слово – "СОН", кожна його літера перетворюється на числове значення: С=21, О=18, Н=17. Отримані значення використовуються для зміщення символів алфавіту під час шифрування чи розшифрування тексту.

14 блок: Формуємо алфавіт на основі обраної мови.

15 блок: Завершення формування таблиці Цезаря.

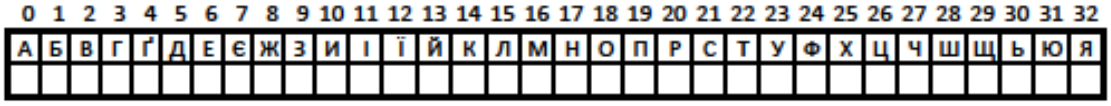


Рисунок 3.2 - Візуалізація українського алфавіту в таблиці Цезаря

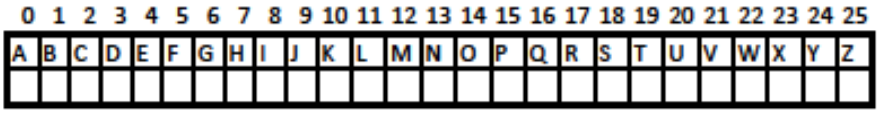


Рисунок 3.3 - Візуалізація англійського алфавіту в таблиці Цезаря

16 блок: Вводимо текст для шифрування, зважаючи на вибраний алфавіт.

17 блок: Заміщуємо символи в тексті за алгоритмом шифрування:

$c = (p + k) \bmod n$

де c – зашифрований символ, p – вихідний символ, k – числовий еквівалент символу ключового слова, n – кількість символів в алфавіті.

18 блок: Записуємо отриману шифрограму .

19 блок: Передача шифрограми користувачу для подальшого використання.

20 блок: Забезпечує можливість почати шифрувати заново для зміни параметрів або повідомлення.

3.2 Розробка схеми електричної структурної системи

Електронна система захисту інформації на базі моноалфавітного алгоритму шифрування призначена для забезпечення необхідного рівня криптостійкості при шифруванні вхідних повідомлень шифром Цезаря з ключовим словом. З цією метою система забезпечує адаптивну зміну параметрів шифрування для кожної частини вхідного повідомлення, визначаючи для такої частини власні ключ та ключове слово. Система, що наведена на рисунку 3.4, складається із наступних блоків, які виконують такі функції:

Блок аналізу вимог до криптостійкості – інформує про необхідний рівень криптостійкості при передачі відповідного повідомлення.

Формувач параметрів шифрування – визначає параметри шифрування у вигляді ключа (кількість символів зсуву між символами алфавіту повідомлення та алфавітом підстановки), ключового слова (фрази або слова), кількості циклів зміни параметрів шифрування для частини повідомлення та кількості таких частин.

Лічильник кількості циклів змін параметрів шифрування – контролює встановлену кількість циклів заміни параметрів шифру та завершення шифрування вхідного повідомлення.

Блок введення повідомлення – забезпечує введення тексту, який потрібно зашифрувати.

Блок визначення алфавіту – використовується для вибору мови, з якою буде виконуватися шифрування.

Сховище алфавітів – містить доступні алфавіти для вибору.

Формувач первинного алфавіту – створює алфавіт відповідно до обраної мови.

Блок введення ключа К – дозволяє ввести значення ключа (К) який визначає зсув символів (де 1<К<Р, а Р-максимальна кількість букв в алфавіті).

Регістр зсуву на К символів – виконує зсув символів алфавіту на задане значення К.

Блок введення ключового слова - забезпечує введення слова, яке визначає правила зсуву символів у таблиці шифрування.

Блок запису ключового слова – записує ключове слово до пам’яті для подальшого використання.

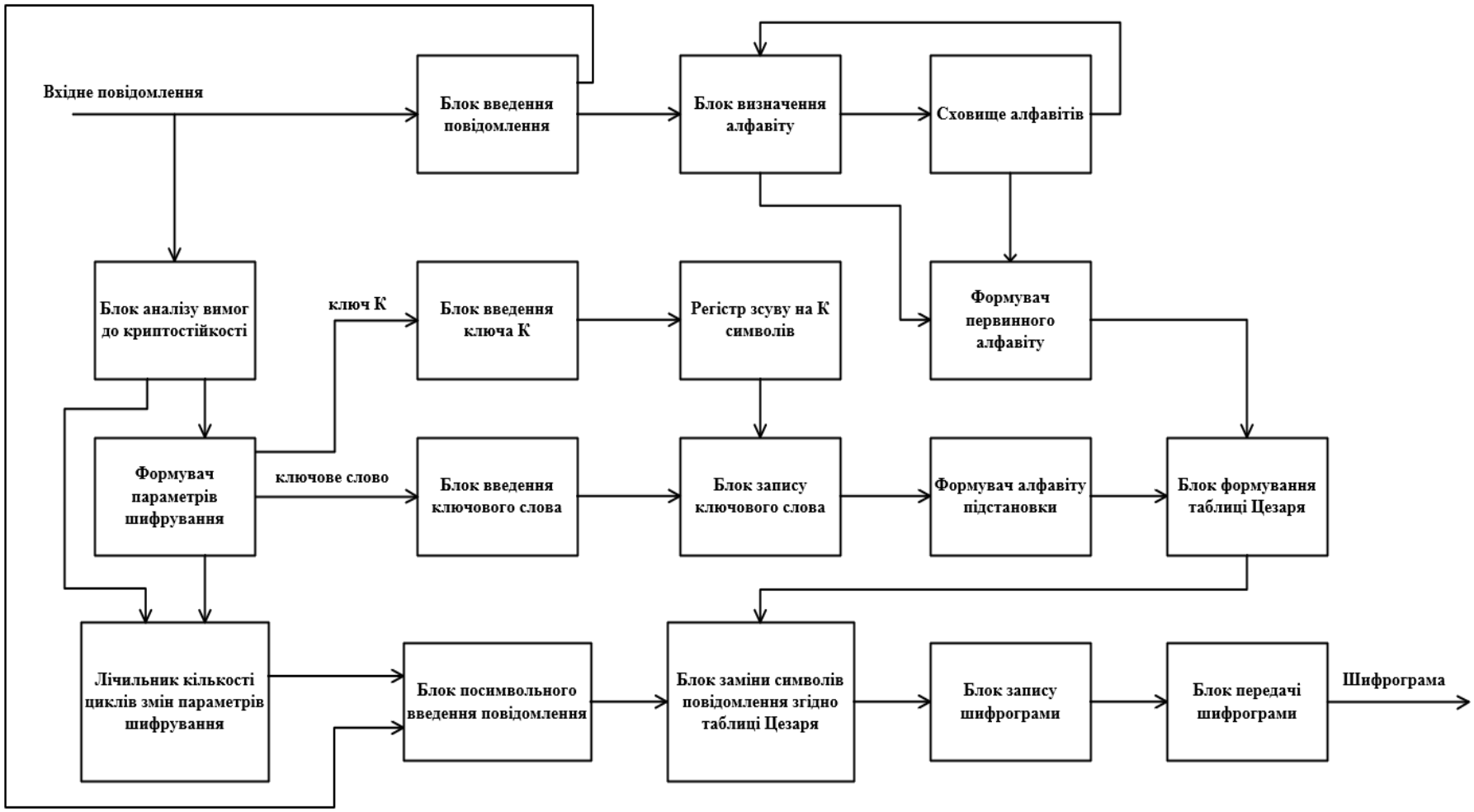


Рисунок 3.4 – Схема електрична структурна

Формувач алфавіту підстановки – генерує алфавіт підстановки на основі ключового слова та зсуву.

Блок формування таблиці Цезаря – створює таблицю Цезаря, яка містить алфавітний рядок із ключовим словом, зміщеним за допомогою ключа K.

Блок по символьного введення повідомлення – дозволяє вводити текст для шифрування посимвольно.

Блок заміни символів повідомлення згідно таблиці Цезаря – виконує заміну символів повідомлення на зашифровані за таблицею.

Блок запису шифрограми – зберігає отриману шифрограмму.

Блок передачі шифрограми – передає шифрограму для подальшого використання та обробки.

4 РОЗРОБКА ФУНКЦІОНАЛЬНОЇ ЕЛЕКТРИЧНОЇ СХЕМИ

Пристрій для захисту інформації, складається з центрального елемента управління, буферних модулів, конвертера сигналів і блоку пам’яті для збереження даних. Його головна мета — забезпечення надійного шифрування текстових повідомлень для їх безпечного зберігання та передачі. Завдяки злагодженій взаємодії компонентів пристрій виконує всі необхідні функції, зокрема обробку, тимчасове зберігання та передачу інформації, забезпечуючи високу ефективність і надійність роботи.

Робота пристрою починається з прийому вхідних даних через вхідний інтерфейс, що слугує точкою зв’язку з зовнішніми системами. Інформація надходить у вигляді сигналів, параметри яких не відповідають стандартам внутрішньої логіки пристрою. Для адаптації сигналів використовується модуль конвертації рівнів, який змінює амплітуду та полярність вхідних сигналів, забезпечуючи їх сумісність із логікою центрального елемента управління.

Цей етап є критично важливим, оскільки будь-які спотворення або несумісності сигналів можуть призвести до втрати інформації. Модуль конвертації також виконує перевірку цілісності сигналу, відкидаючи шумові компоненти.

Прийнята інформація передається до центрального елемента управління, де запускається основний алгоритм шифрування. Цей процес включає виконання послідовних операцій заміни символів згідно з заздалегідь визначеним ключем. Центральний елемент управління виконує аналіз вхідних даних, розбиваючи їх на символи, які обробляються покроково.

Для зберігання проміжних результатів обчислень застосовуються буферні модулі. Вони забезпечують тимчасову пам’ять для символів, які вже пройшли частину обробки, але ще не готові до запису у зовнішню пам’ять або передачі. Буфери зменшують навантаження на центральний елемент управління, дозволяючи йому виконувати інші операції паралельно.

Проміжні результати зберігаються у буфері доти, доки не завершиться повний цикл обробки вхідного повідомлення. Така організація роботи забезпечує синхронізацію між етапами шифрування та мінімізує ризик втрати даних через можливі збої.

Після завершення обробки результати шифрування записуються у зовнішній блок збереження даних. Цей модуль є енергонезалежним та має високу швидкодію, що дозволяє швидко зберігати великі обсяги інформації.

Зовнішній блок пам’яті взаємодіє з центральним елементом управління через адресну шину, яка вказує місце запису або зчитування даних. Шина даних передає безпосередньо зашифровану інформацію, тоді як сигнали управління визначають режим роботи пам’яті — читання чи запис.

Розподіл інформації у зовнішній пам’яті організовано таким чином, щоб забезпечити швидкий доступ до даних. Це особливо важливо, коли пристрій працює в режимі реального часу.

Якщо пристрій налаштований на передачу зашифрованої інформації, збережені в пам’яті дані зчитуються, проходять через буферний модуль і передаються до вихідного інтерфейсу. Буфер забезпечує стабільну передачу сигналів, усуваючи асинхронність між пам’яттю та вихідним інтерфейсом.

Перед тим як дані покинуть пристрій, модуль конвертації знову адаптує рівні сигналів до вимог зовнішніх систем. Завдяки цьому забезпечується сумісність із приймальними пристроями.

Протягом усієї роботи пристрою центральний елемент управління виконує безперервний контроль стану всіх компонентів. Він отримує сигнали про готовність від кожного модуля, аналізує їх і приймає рішення про наступні дії.

У разі виявлення несправностей, наприклад, збою в зовнішній пам’яті чи втрати сигналу на вихідному інтерфейсі, центральний елемент ініціює повторне виконання операції. Це дозволяє мінімізувати вплив збоїв на загальну роботу пристрою.

Схема електрична функціональна пристрою наведена на рисунку 4.1.

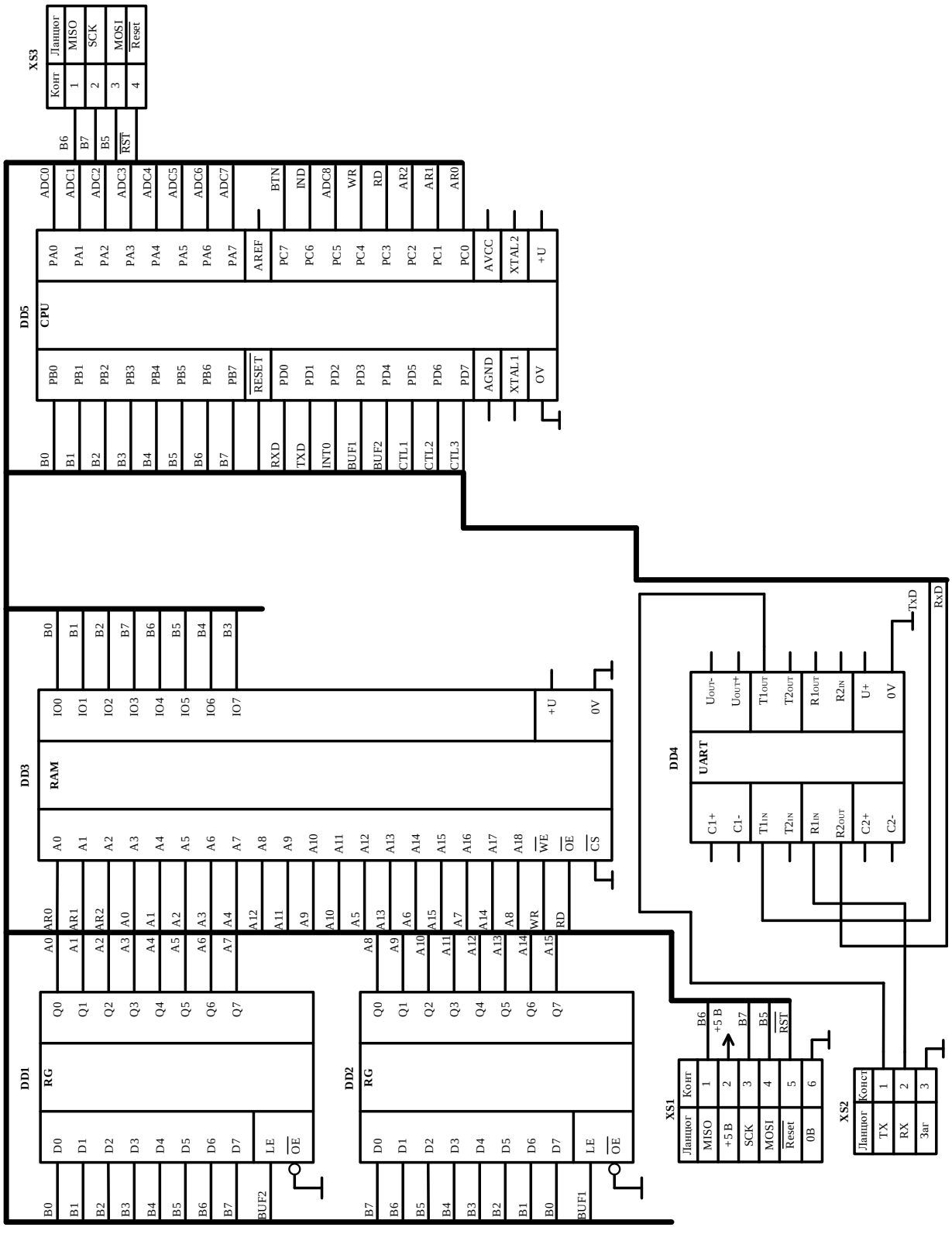


Рисунок 4.1 – Схема електрична функціональна

5 РОЗРОБКА СХЕМИ ЕЛЕКТРИЧНОЇ ПРИНЦИПОВОЇ

5.1 Мікроконтролер ATMEGA8535

Мікроконтролер ATMEGA 8535 є центральним елементом пристрою шифрування інформації, що реалізує алгоритм Цезаря з ключовим словом. Основна його функція — забезпечення управління всіма процесами в системі. Він приймає вхідні дані через порти введення-виведення, обробляє їх за заданим алгоритмом шифрування і передає результати до вихідних пристроїв. Мікроконтролер виконує роль "мозку" пристрою, оскільки об'єднує всі компоненти в єдину функціональну систему.

Цей мікроконтролер працює у тісній взаємодії з пам'яттю, інтерфейсними мікросхемами та іншими периферійними елементами. З його допомогою реалізується генерація тактових сигналів, необхідних для синхронізації всіх компонентів пристрою. Він забезпечує взаємодію з оперативною пам’яттю K6T4008C1B-GB55, де тимчасово зберігаються дані, що обробляються. Через інтерфейс RS-232, який реалізується за допомогою мікросхеми MAX232EPE, мікроконтролер спілкується із зовнішніми пристроями, такими як комп’ютери чи термінали.

Важливим аспектом роботи ATMEGA 8535 є його здатність керувати паралельними шинами даних, використовуючи TTL-мікросхему 74HKT573, яка служить для буферизації сигналів та управління напрямком передачі. Цей мікроконтролер також виконує функції шифрування, реалізуючи обчислення, необхідні для перетворення тексту згідно з методом Цезаря. Завдяки програмованим входам-виходам він може адаптуватися до різних конфігурацій пристрою.

ATMEGA 8535 підтримує широке коло інтерфейсів, таких як UART, SPI та I²C, що робить його універсальним і дозволяє використовувати в різних проєктах. У цьому пристрої він не лише відповідає за основні обчислення, але й виконує функцію координації дій усіх компонентів, забезпечуючи надійне та узгоджене функціонування системи. Завдяки високій енергоефективності та простоті програмування, цей мікроконтролер ідеально підходить для реалізації задач криптографії та обробки даних у реальному часі.

Перелік функцій які виконує ATMEGA8535:

Головною функцією мікроконтролера є виконання програмного коду, записаного у вбудованій Flash-пам’яті. Для шифрування тексту за методом Цезаря він опрацьовує символи тексту, зміщуючи їх у відповідності до значення ключового слова. Алгоритм обробки включає циклічні обчислення, умовні оператори та роботу з табличними даними. Процесорний блок відповідає за обробку інструкцій та виконує команди з використанням вбудованих регістрів, що забезпечує високу швидкість роботи.

ATMEGA 8535 має кілька типів вбудованої пам'яті для різних завдань:

•Flash-пам’ять обсягом 8 КБ використовується для зберігання програмного коду.

•SRAM (512 Б) слугує оперативною пам’яттю, де тимчасово зберігаються дані під час обробки.

•EEPROM (512 Б) дозволяє зберігати постійні налаштування, такі як ключове слово для шифрування або параметри конфігурації.

Крім того, мікроконтролер здійснює обмін даними з зовнішньою статичною RAM K6T4008C1B-GB55, використовуючи адресні та дані шини для тимчасового збереження оброблюваних текстових даних.

Мікроконтролер відповідає за координацію роботи підключених компонентів через свої порти введення-виведення. Використовуючи TTL-мікросхему 74HKT573, він керує передачею даних між внутрішньою пам’яттю та іншими вузлами системи. Ця мікросхема забезпечує буферизацію сигналів і перетворення рівнів, що дозволяє коректно керувати периферією.

ATMEGA 8535 має кілька інтерфейсів зв'язку для інтеграції з іншими пристроями:

•UART (Universal Asynchronous Receiver-Transmitter) використовується для обміну даними через RS-232 за допомогою мікросхеми MAX232EPE, яка перетворює рівні сигналів на стандартні для комп'ютерних систем. Це дозволяє передавати шифровані дані або приймати команди ззовні.

•SPI (Serial Peripheral Interface) та I²C (Inter-Integrated Circuit) слугують для швидкого обміну даними між мікроконтролером і додатковими периферійними пристроями, якщо це необхідно.

Мікроконтролер має 32 програмовані виводи введення-виведення, які можуть використовуватися для прийому та передачі цифрових сигналів. Ці порти

дозволяють взаємодіяти з зовнішніми пристроями, наприклад, клавіатурою або індикатором для введення-виведення тексту.

Для стабільної роботи системи ATMEGA 8535 використовує зовнішній кварцовий резонатор з частотою 6.4 МГц, що забезпечує синхронізацію обчислень і роботу периферійних пристроїв.

ATMEGA 8535 має кілька режимів енергоспоживання, включаючи сплячий режим, який активується, коли пристрій не виконує обчислення. Це дозволяє значно зменшити енергоспоживання системи.

Мікроконтролер оснащений 10-бітовим аналого-цифровим перетворювачем (ADC) з вісьмома каналами, що дозволяє працювати з аналоговими сигналами, якщо це потрібно для розширення функціоналу пристрою.

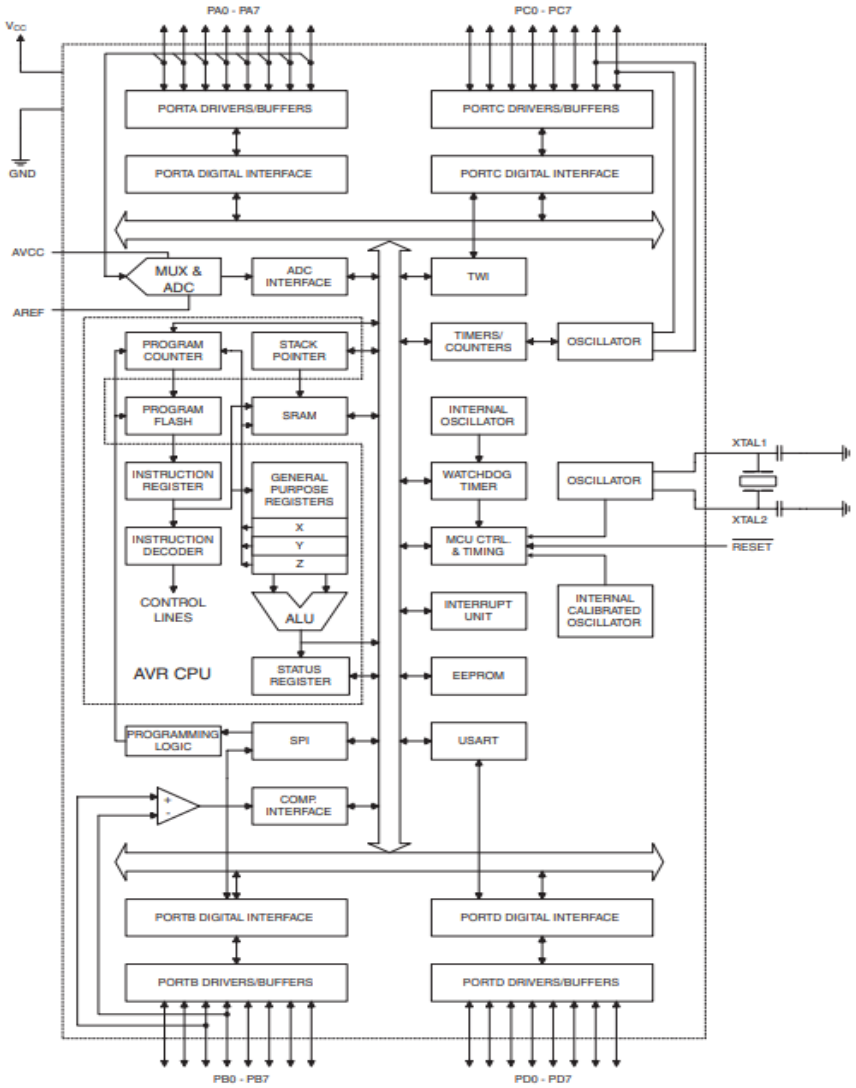


Рисунок 5.1 – Функціональна блок-схема мікроконтролеру ATMEGA 8535

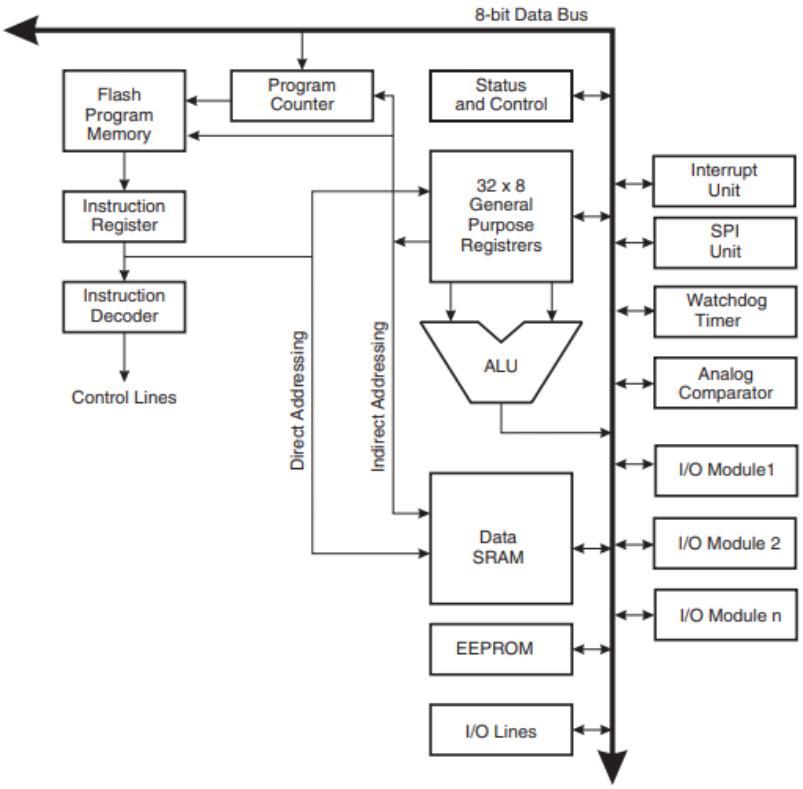


Рисунок 5.2 – Структурна блок-схема мікроконтролеру ATMEGA 8535

Структурна блок-схема має наступні блоки та їх застосування:

Flash Program Memory: Зберігає програму, яку виконує мікроконтролер. Програма складається з послідовності команд, які виконуються процесором.

Program Counter: Вказує на адресу наступної команди, яка буде виконана.

Status and Control: Регістр, який зберігає інформацію про стан процесора (прапорці) та використовується для керування різними функціями мікроконтролера.

General Purpose Registers: Набір регістрів загального призначення, які використовуються для тимчасового зберігання даних під час виконання програми.

Instruction Register: Зберігає поточну команду, яка виконується процесором.

Instruction Decoder: Декодує інструкції, тобто перетворює їх на послідовність елементарних операцій, які можуть бути виконані процесором.

ALU (Arithmetic Logic Unit): Виконує арифметичні та логічні операції над даними.

Data SRAM: Статична оперативна пам'ять, використовується для зберігання даних, які часто змінюються під час виконання програми.

EEPROM: Енергонезалежна пам'ять, використовується для зберігання даних, які повинні зберігатися навіть після вимкнення живлення.

I/O Modules: Забезпечують взаємодію мікроконтролера з зовнішніми пристроями через вхідні та вихідні лінії.

Interrupt Unit: Обробляє переривання, що дозволяє мікроконтролеру реагувати на зовнішні події.

SPI Unit: Серійний периферійний інтерфейс, використовується для обміну даними з іншими пристроями.

Watchdog Timer: Таймер, який використовується для перезапуску мікроконтролера у випадку зависання.

Analog Comparator: Порівнює два аналогових сигнали.

Принцип дії:

1. Завантаження програми - Програма завантажується в пам'ять Flash.
2. Виконання програми - Процесор починає виконувати програму з першої команди.
3. Фетч цикл - Процесор зчитує інструкцію з пам'яті Flash за адресою, вказаною в лічильнику команд.
4. Декодування - Інструкція декодується, і процесор визначає, яку операцію потрібно виконати.
5. Виконання - Процесор виконує необхідні операції, використовуючи ALU, регістри загального призначення та інші компоненти.
6. Зміна лічильника команд - Після виконання команди лічильник команд збільшується на розмір інструкції, і процесор переходить до виконання наступної команди.
7. Переривання - Якщо відбувається переривання, процесор призупиняє виконання поточної програми і переходить до обробки переривання. Після обробки переривання процесор повертається до виконання початкової програми.

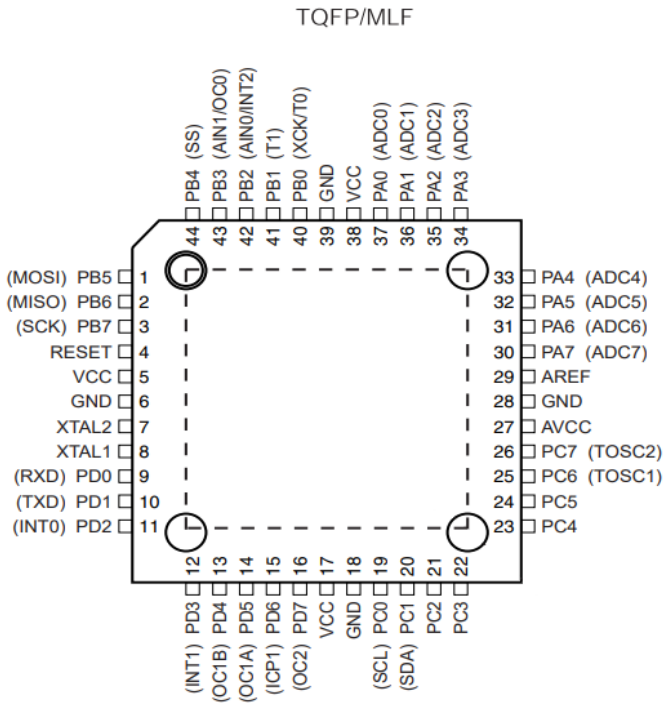


Рисунок 5.3 – Зображення мікроконтролера ATMEGA8535 в корпусі TQFP та MLF

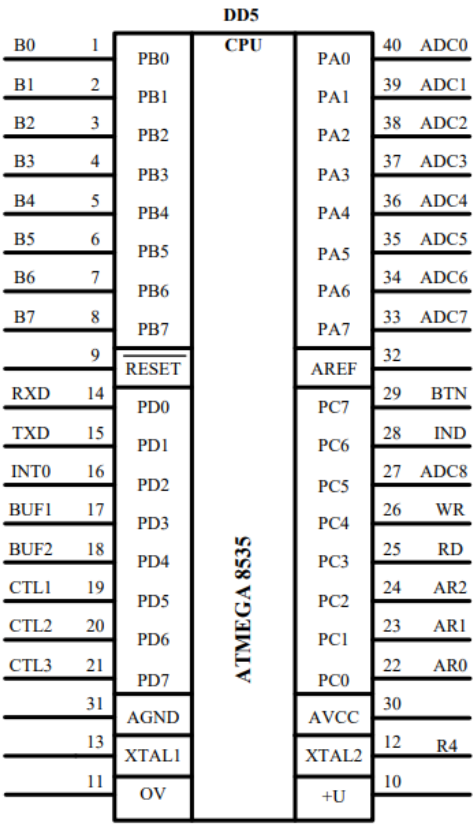


Рисунок 5.4 – Зображення принципової схеми ATMEGA 8535

Таблиця 5.1 – Входи/виходи мікроконтролера ATMEGA8535

Позначення	Номер виводу	Тип	Опис
XTAL1	13	I	Вхід тактового генератора
XTAL2	12	O	Вихід тактового генератора
RESET	9	I	Вхід збросу
Виводи живлення			
AREF	32	P	Вхід опорної напруги для АЦП
AVCC	30	P	Виведення джерела живлення АЦП
VCC	10	P	Виведення джерела живлення
GND	11	P	Загальний вихід
Порт A. 8-бітний двонаправлений порт вводу-виводу з внутрішніми підтягуючими резисторами			
PA0	40	I\O	0-й біт порту A. Вхід АЦП
PA1	39	I\O	1-й біт порту A. Вхід АЦП
PA2	38	I\O	2-й біт порту A. Вхід АЦП
PA3	37	I\O	3-й біт порту A. Вхід АЦП
PA4	36	I\O	4-й біт порту A. Вхід АЦП
PA5	35	I\O	5-й біт порту A. Вхід АЦП
PA6	34	I\O	6-й біт порту A. Вхід АЦП
PA7	33	I\O	7-й біт порту A. Вхід АЦП
Порт B. 8-бітний двонаправлений порт вводу-виводу з внутрішніми підтягуючими резисторами			
PB0	1	I\O	0-й біт порту B Вхід зовнішнього тактового сигналу таймера/лічильника T0 Вхід/вихід зовнішнього тактового сигналу USART
PB1	2	I\O	1-й біт порту B. Вхід зовнішнього тактового сигналу таймера/лічильника T1

Продовження таблиці 5.1

PB2	3	I\O	2-й біт порту B Неінвертуючий вхід компаратора. Вхід зовнішнього переривання
PB3		I\O	3-й біт порту B Інвертуючий вхід компаратора Вихід таймера-лічильника T0
PB4	4	I\O	4-й біт порту B Вибір Slave-пристрою на шині SPI
PB5	5	I\O	5-й біт порту B. Вихід (Master) або вхід (Slave) даних модуля SPI
PB6	6	I\O	6-й біт порту B. Вихід (Master) або вхід (Slave) даних модуля SPI
PB7	7	I\O	7-й біт порту B. Вихід (Master) або вхід (Slave) Тактового сигналу модуля SPI
Порт C. 8-бітний двонаправлений порт вводу-виводу з внутрішніми підтягуючими резисторами			
PC0	22	I\O	0-й біт порту C. Вхід/вихід тактового сигналу модуля TWI
PC1	23	I\O	1-й біт порту C. Вхід/вихід даних модуля TWI
PC2	24	I\O	2-й біт порту C
PC3	25	I\O	3-й біт порту C
PC4	26	I\O	4-й біт порту C
PC5	27	I\O	5-й біт порту C
PC6	28	I\O	6-й біт порту C Вивід для підключення резонатора до таймеру/лічильнику T2

Продовження таблиці 5.1

PC7	29	I\O	7-й біт порту C Вивід для підключення резонатора до таймера- лічильнику T2
Порт D. 8-бітний двонаправлений порт вводу-виводу з внутрішніми підтягуючими резисторами			
PD0	14	I\O	0-й біт порту D. Вхід USART
PD1	15	I\O	1-й біт порту D. Вихід USART
PD2	16	I\O	2-й біт порту D. Вхід зовнішнього переривання
PD3	17	I\O	3-й біт порту D. Вхід зовнішнього переривання
PD4	18	I\O	4-й біт порту D. Вихід B таймера-лічильника T1
PD5	19	I\O	5-й біт порту D. Вихід A таймера-лічильника T1
PD6	20	I\O	6-й біт порту D. Вхід захоплення таймера-лічильника T1
PD7	21	I\O	7-й біт порту D. Вихід таймера-лічильника T2

Часові діаграми взаємодії вхідних та вихідних сигналів:

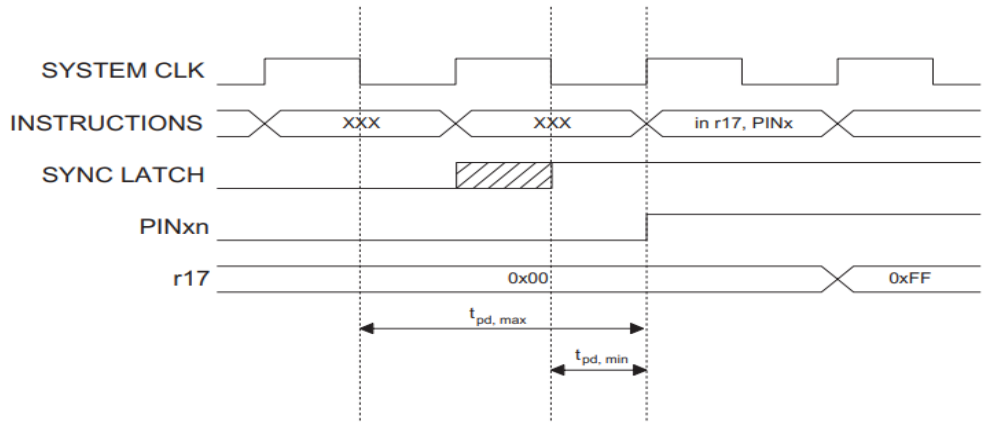


Рисунок 5.5 - Синхронізація при зчитуванні зовнішнього значення контакту

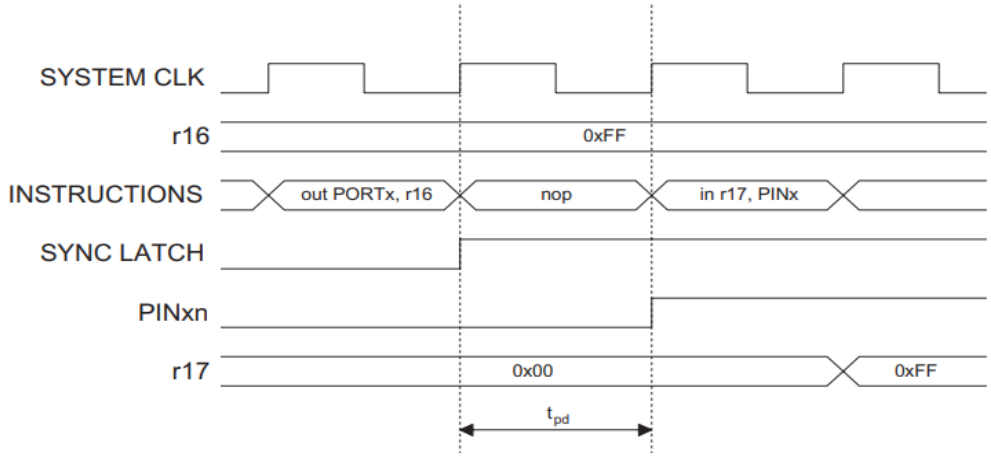


Рисунок 5.6 - Синхронізація при зчитуванні призначеного програмним забезпеченням значення контакту

5.2 Тригер 74HCT573

74HCT573 — це 8-бітний тригер типу D із фіксацією стану на виходах і трьохстановими виходами. Її головне призначення — тимчасове зберігання даних і керування потоком даних у цифрових системах. Використовується для буферизації сигналів у складних мікропроцесорних системах, де необхідно забезпечити стабільність сигналів та їх розділення між різними шинами даних.

Він використовується для передачі сигналів від одного модуля до іншого в моменти часу, що строго визначаються сигналом керування. Завдяки трьохстановим виходам (активний високий, активний низький або відключений), дозволяє підключати кілька пристроїв до спільної шини даних без конфліктів. Забезпечує фіксацію стану даних, що дозволяє системі працювати в умовах різної швидкості передачі сигналів між компонентами.

Використовується для буферизації та розширення шин даних, синхронізації між компонентами із різною швидкістю передачі інформації, а також для ізоляції сигналів у складних схемах. 74HCT573 забезпечує буферизацію даних між мікроконтролером ATMEGA 8535 та іншими компонентами. Вона працює як фіксатор даних, який утримує значення на виходах до моменту їх зчитування, синхронізуючи процеси запису і читання між різними частинами системи.

Основні функції 8-бітного тригеру включають тимчасове зберігання даних, синхронізацію сигналів, керування потоками даних на загальних шинах і запобігання конфліктам між компонентами.

Перша важлива функція 74HCT573 — це буферизація даних. Вона зберігає значення на своїх виходах до моменту, коли це буде потрібно для передачі чи подальшої обробки. Це особливо корисно в системах, де дані передаються асинхронно, і потрібно точно визначити момент їх фіксації для подальшої роботи.

Другою важливою функцією є керування напрямком потоку даних між різними елементами системи. Завдяки трьохстановим виходам, тригер дозволяє на вимогу відключати виходи, що робить можливим підключення кількох пристроїв до однієї шини без ризику конфліктів. Це дозволяє організувати мультиплексування даних і підвищити ефективність системи за рахунок спільного використання ліній зв'язку для кількох пристроїв.

Ще однією важливою задачею є синхронізація сигналів в умовах багатозадачності. У складних системах із кількома мікросхемами або мікроконтролерами, що працюють із різними частотами такту, потрібна точна синхронізація між різними компонентами. 74HCT573 допомагає вирішити цю задачу, чітко фіксуючи і зберігаючи дані на виходах до того, як наступна операція на шині буде завершена, забезпечуючи тим самим узгодженість сигналів на входах і виходах.

Також 74HCT573 може виконувати функцію розширення кількості виходів. Оскільки вона має 8 виходів, ці лінії можуть бути використані для додаткових сигналів у схемах, де мікроконтролер чи інший центральний процесор має обмежену кількість доступних виводів. Це дозволяє істотно зменшити потребу в додаткових мікросхемах, зберігаючи компактність і спрощуючи конструкцію системи.

Можлива зміна стану сигналів за допомогою зовнішнього керуючого сигналу. Сигнал LE (Latch Enable) дозволяє змінювати стан виходів, а сигнал OE (Output Enable) контролює, чи будуть виходи підключені до шини або відключені.

74HCT573 забезпечує стабільну та ефективну роботу, виконує функції буферизації, синхронізації, керування напрямком потоку даних і розширення кількості виходів.

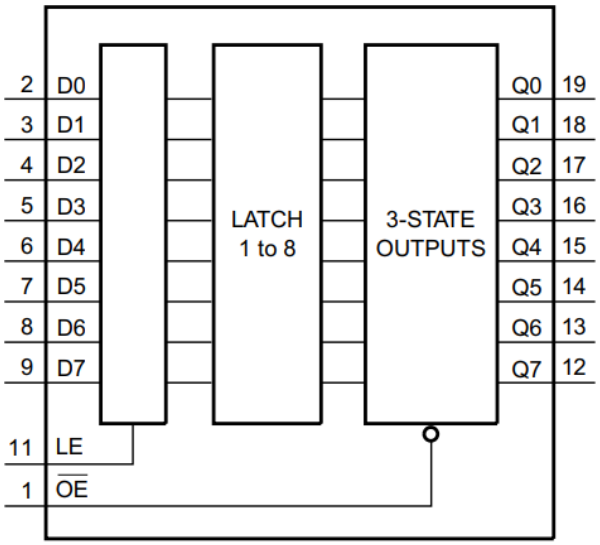


Рисунок 5.7 – Зображення функціональної схеми 74HCT573

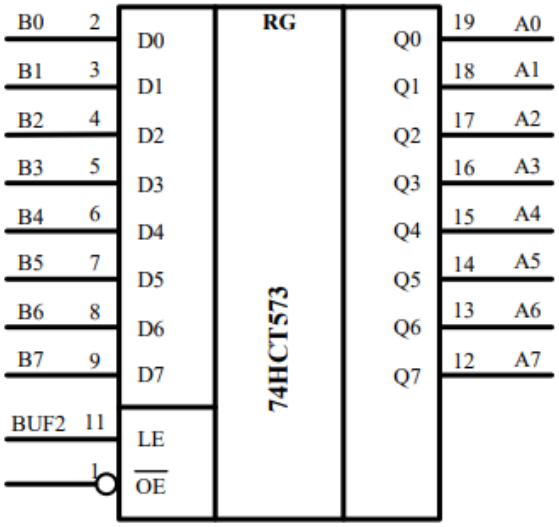


Рисунок 5.8 – Зображення принципової схеми 74HCT573

Таблиця 5.2 - Входи/виходи 74HCT573

Позначення	Номер виводу	Опис
OE	1	Вхід для включення 3-станного виходу (активний LOW)
D0 – D7	2,3,4,5,6,7,8,9	Введення даних
GND	10	Заземлення
LE	11	Вхід для включення засувки (активний HIGH)

Продовження таблиці 5.2

Q0 – Q7	19,18,17,16,15,14,13,12	3-становий вхід напруги
Vcc	20	Напруга живлення

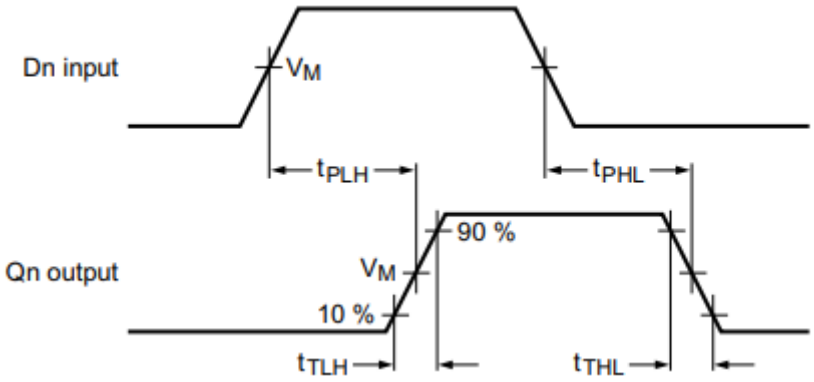


Рисунок 5.9 - Затримка поширення вхідних даних (Dn) на вихід (Qn) і час переходу на виході

74HCT573 має наступні характеристики:

Напруга живлення - 4.5V до 5.5V.

Потужність споживання залежить від умов роботи, але в основному цей тригер має низьке споживання енергії, характерне для серії HCT (High-speed CMOS). Типове споживання становить кілька мікроампер при роботі в режимі очікування, що дозволяє знизити енергоспоживання у більшості цифрових систем.

Рівень напруги сигналів:

- Вхідні сигнали (D0-D7, LE, OE): Від 0 до 5V (логіка рівня "high" — від 2V до 5V, "low" — від 0 до 0.8V).
- Вихідні сигнали (Q0-Q7): Логіка рівня 0 до 5V, з відповідними характеристиками для "high" (2V–5V) і "low" (0–0.8V). Виходи здатні працювати в режимі трьох станів.

Має 8 виходів (Q0-Q7), які можуть працювати в трьох режимах: високий рівень, низький рівень або високий імпеданс (вимкнено). Це дозволяє з'єднувати кілька мікросхем із загальною шиною без конфліктів.

Тригер не має вбудованої пам'яті в класичному розумінні (як флеш-пам'ять чи SRAM), оскільки її основне завдання — зберігання даних на виходах при активації сигналу LE.

74HCT573 має паралельний інтерфейс для взаємодії з іншими цифровими мікросхемами та системами через 8 біт входів (D0-D7) та виходів (Q0-Q7). Виходи можуть бути переведені в трьохвихідний стан завдяки сигналу OE.

5.3 Перетворювач MAX232EPE

MAX232EPE —призначена для перетворення рівнів напруги сигналів між логікою TTL і стандартами RS-232. Вона забезпечує двосторонню передачу даних за допомогою порту серійного зв'язку.

Містить два трансивери, які дозволяють перетворювати логічні рівні TTL (0–5V) у відповідні рівні сигналів для RS-232 (від ±12V до +12V), і навпаки. Це дозволяє з’єднувати мікроконтролери або цифрові пристрої, які працюють з логікою 0–5V, з пристроями, що використовують стандарт RS-232 для серійної передачі даних, де рівні сигналів можуть бути набагато вищими.

Основне призначення — забезпечення правильної взаємодії між цифровими мікропроцесорними системами та устаткуванням, яке використовує RS-232 для обміну даними, наприклад, при підключенні до комп'ютера через COM-порт або при організації зв’язку між різними мікроконтролерами та периферійними пристроями.

MAX232EPE забезпечує надійне і ефективне перетворення сигналів, що є критичним для правильної роботи у системах, де потрібно взаємодіяти з комп'ютерними терміналами або іншими серійними пристроями.

MAX232EPE виконує декілька ключових функцій, забезпечуючи перетворення рівнів напруги між різними стандартами сигналів для передачі даних. Основні функції цієї мікросхеми включають:

1. Перетворення сигналів RS-232 в TTL і навпаки. Основною функцією MAX232EPE є перетворення напруги сигналів з високих рівнів стандарту RS-232 (±12V) у низькі рівні TTL (0–5V), а також навпаки — перетворення логіки TTL у стандарти RS-232. Цей процес є критичним для зв'язку між сучасними цифровими пристроями, такими як мікроконтролери, і пристроями, що використовують традиційний серійний порт RS-232.

2. Двосторонній зв'язок. Перетворювач підтримує двосторонню передачу даних, що дозволяє не лише приймати сигнали з зовнішнього пристрою, але й відправляти їх назад. Завдяки цьому забезпечується повна взаємодія між

пристроями через серійний інтерфейс, що особливо важливо для обміну даними в реальному часі.

3. Ізоляція рівнів сигналів. MAX232EPE здійснює ізоляцію між логікою низької напруги (TTL) і пристроями, які використовують високі напруги (RS-232). Це дозволяє уникнути пошкоджень мікросхем або мікроконтролерів через надмірну напругу та підвищує безпеку роботи пристроїв.

4. Підтримка кількох ліній зв'язку. Перетворювач оснащений двома приймачами та двома передавачами, що дозволяє їй обробляти два канали серійного зв'язку одночасно. Це дозволяє з'єднувати один мікроконтролер з кількома пристроями або створювати багатоканальну серійну комунікацію, якщо потрібно.

5. Стабільність і надійність сигналів. MAX232EPE забезпечує стабільну передачу сигналів через довгі кабелі або в умовах електричних завад завдяки використанню вбудованих фільтрів та регулювання рівнів сигналів, що важливо для надійної роботи в промислових і побутових додатках.

6. Підтримка низького енергоспоживання. Перетворювач розроблений так, щоб споживати мінімальну кількість енергії, що дозволяє йому використовувати в енергозалежних системах, таких як портативні пристрої або пристрої з низьким енергоспоживанням.

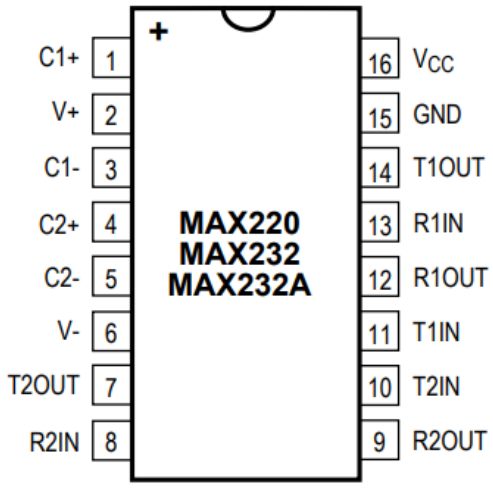


Рисунок 5.10 – Зображення перетворювача MAX232EPE

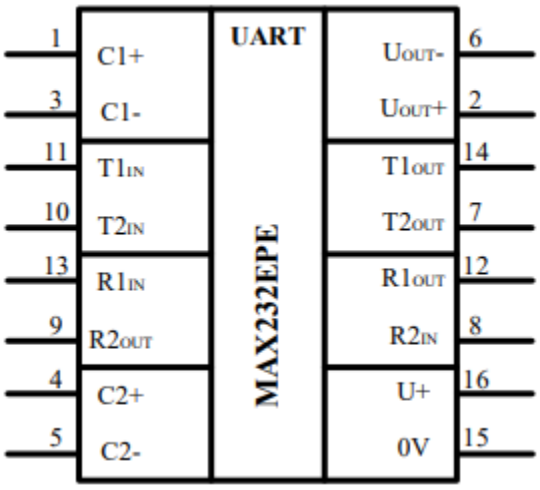


Рисунок 5.11 - Зображення перетворювача MAX232EPE

Таблиця 5.3 – Входи/виходи MAX232EPE

Позначення	Номер виводу	Опис
Vcc	16	Живлення
GND	15	Заземлення
C1+	1	Позитивні виводи конденсатора для підвищення напруги
C2+	4	
C1-	3	Негативні виводи конденсатора для підвищення напруги
C2-	5	
V+	2	Подача високої та низької напруги для підвищення/пониження рівня сигналів відповідно до стандарту RS-232.
V-	6	
T1IN	11	Вхід для прийому сигналів від порту RS-232
T2IN	10	
T1OUT	14	Виводи для передачі сигналів до порту RS-232
T2OUT	7	
R1IN	13	Вхід для прийому сигналів від порту RS-232
R2IN	8	
R1OUT	12	Виводи для передачі сигналів до логіки TTL
R2OUT	9	

Часові діаграми:

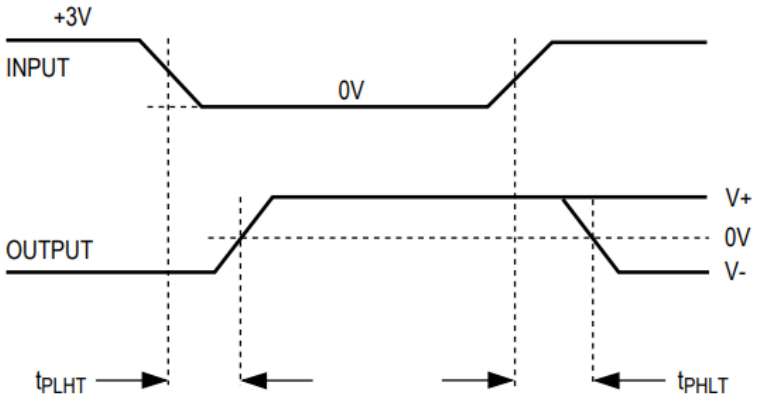


Рисунок 5.12 – Час поширення та затримки передавача

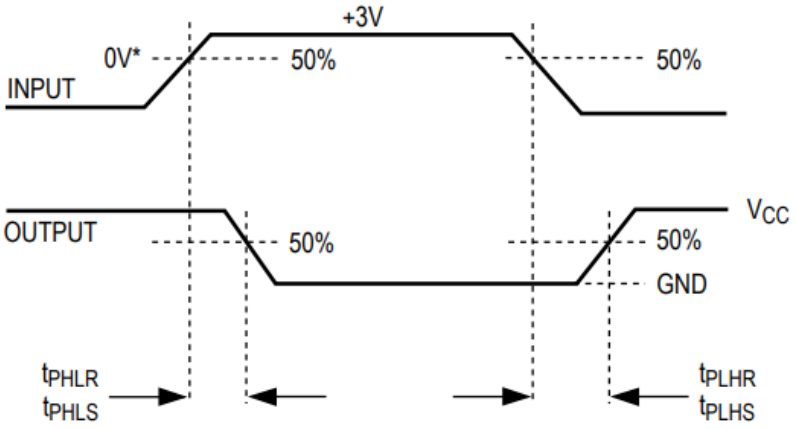


Рисунок 5.13 – Синхронізація поширення та затримки приймача

Характеристики перетворювача:

Напруга живлення перетворювача:

Vcc: 5 В ±10%.

Споживана потужність залежить від умов роботи та навантаження, але типово становить <1 Вт при максимальному струмі.

Рівень напруги сигналів перетворювача:

•Вхідні сигнали (TTL): Вхідні сигнали повинні відповідати стандарту TTL (0 до 5 В).

•Вихідні сигнали (RS-232): Виводи сигналів для RS-232 можуть мати рівні напруги ±12 В для передачі сигналів по порту RS-232.

•Для MAX232EPE типові рівні напруги для логіки TTL: логічний "0" — від 0 до 0.8 В, логічний "1" — від 2 до 5 В.

У перетворювача є два канали для передачі та прийому даних, що дозволяє передавати сигнали в двох напрямках одночасно. Проте точні цифри коефіцієнта розгалуження не зазначені в даташиті.

MAX232EPE не має вбудованої пам'яті, оскільки вона виконує лише функцію перетворення сигналів між стандартами RS-232 та TTL.

Типи інтерфейсів:

• RS-232: Інтерфейс для зв'язку з іншими пристроями, що використовують стандарт RS-232 (наприклад, комп'ютери, модеми).

• TTL (Transistor-Transistor Logic): Інтерфейс для зв'язку з цифровою логікою (мікроконтролерами, мікропроцесорами та іншими компонентами, що використовують стандарт TTL).

5.4 Пам’ять K6T4008C1B-GB55

K6T4008C1B-GB55 є чіпом пам'яті типу SRAM (статична пам'ять з випадковим доступом). Вона використовується для тимчасового зберігання даних у цифрових пристроях, таких як мікроконтролери, мікропроцесори та інші електронні системи. Це статична пам'ять, що не потребує періодичного оновлення (відмінно від динамічної пам'яті), що робить її більш швидкою та енергозберігаючою у порівнянні з іншими типами пам'яті, такими як DRAM (динамічна пам'ять).

Елемент має 8-розрядну організацію пам'яті з об'ємом 512 Кб (кило-байт), що забезпечує високу швидкість доступу до збережених даних, і є основною для реалізації оперативної пам'яті в пристроях, де швидкість читання та запису критична. Ключовим її застосуванням є зберігання тимчасових даних, таких як регістри або буфери в електронних системах, що виконують обчислення в реальному часі або працюють з великими обсягами інформації.

Завдяки своїй організації, K6T4008C1B-GB55 є важливим компонентом для швидкого доступу до даних в системах, де потрібно часто звертатися до пам'яті, а також забезпечує високу надійність при збереженні інформації в умовах короткочасних перерв у живленні або інших зовнішніх впливів.

K6T4008C1B-GB55 є DRAM типом пам'яті і виконує основну функцію зберігання даних в цифрових системах, зокрема для обробки та тимчасового зберігання інформації, яка швидко змінюється. Ось основні функції, які виконує ця мікросхема:

Зберігання даних: Основна функція K6T4008C1B-GB55 — це зберігання цифрових даних. Мікросхема забезпечує високошвидкісне зберігання бітових даних (1 або 0) в своїх осередках, що використовуються для обробки інформації в різних цифрових системах. Це забезпечує тимчасове зберігання даних, необхідних для виконання обчислень чи операцій.

Швидкий доступ до даних: Забезпечує швидкий доступ до збережених даних для подальшої обробки чи передачі. Це дозволяє прискорити виконання операцій, таких як пошук або оновлення значень в обчислювальних процесах.

Динамічне оновлення даних: Як типова DRAM пам'ять, K6T4008C1B-GB55 вимагає періодичної ревізії або оновлення своїх осередків пам'яті для підтримки актуальності збережених даних. Це дозволяє забезпечити стабільність і коректність збережених значень.

Інтерфейс з іншими компонентами: Взаємодіє з процесорами чи іншими мікросхемами через стандартні інтерфейси пам'яті, такі як шина даних, що дозволяє інтегрувати її в більші системи для зберігання великих обсягів даних.

Підтримка низького споживання енергії: Хоча DRAM мікросхеми зазвичай споживають більше енергії, ніж інші типи пам'яті, K6T4008C1B-GB55 оптимізовано для зменшення енергоспоживання в режимі очікування, що підвищує ефективність використання енергії в системах, де ця пам'ять застосовується.

Сумісність з широким спектром систем: Оскільки ця пам'ять є частиною сімейства DRAM, вона може бути інтегрована в різноманітні комп'ютерні та вбудовані системи, забезпечуючи гнучкість при проектуванні обчислювальних платформ або пристроїв, що потребують великої кількості пам'яті для обробки даних.

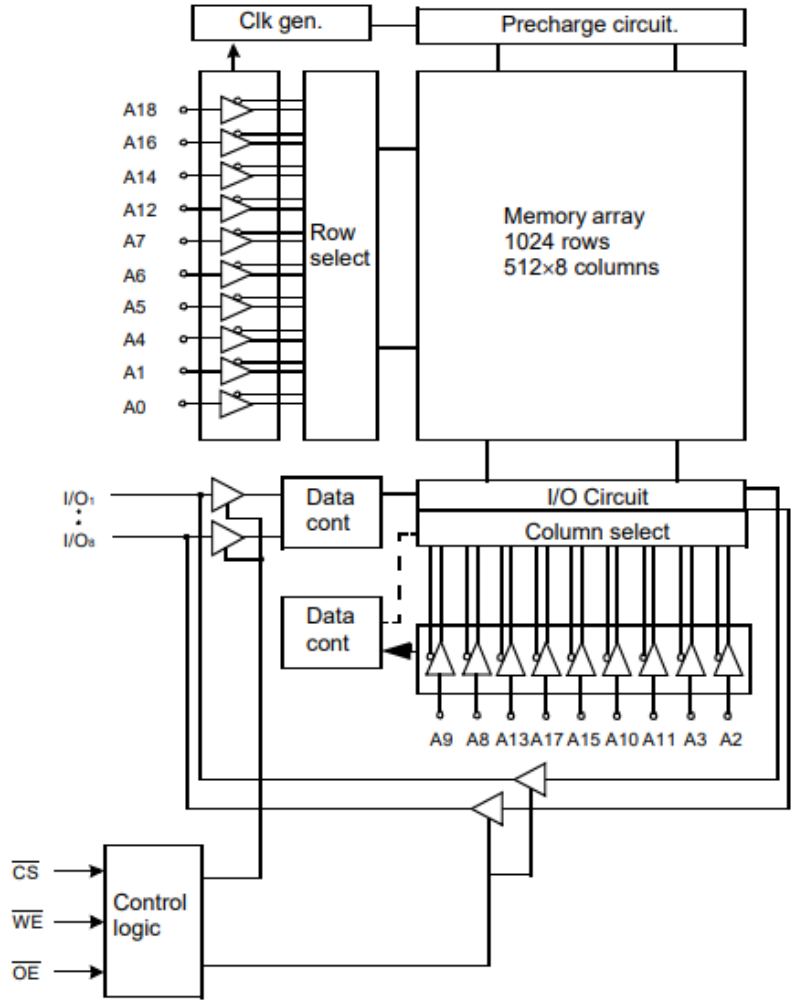


Рисунок 5.14 – Функціональна блок-схема статичної оперативної пам'яті K6T4008C1B-GB55

Опис блоків функціональної блок-схеми (рис. 5.15):

Clk gen (Генератор тактових імпульсів): забезпечує синхронізацію роботи всієї схеми. Він генерує сигнали такту, які синхронізують всі операції пам'яті, забезпечуючи коректне зчитування та запис даних.

Precharge circuit (Прехедж-схема): виконує функцію попереднього заряджання всіх ліній стовпців пам'яті перед тим, як здійснити операцію зчитування або запису. Це необхідно для забезпечення коректного стану осередків пам'яті перед виконанням операцій, що дозволяє уникнути помилок при доступі до даних.

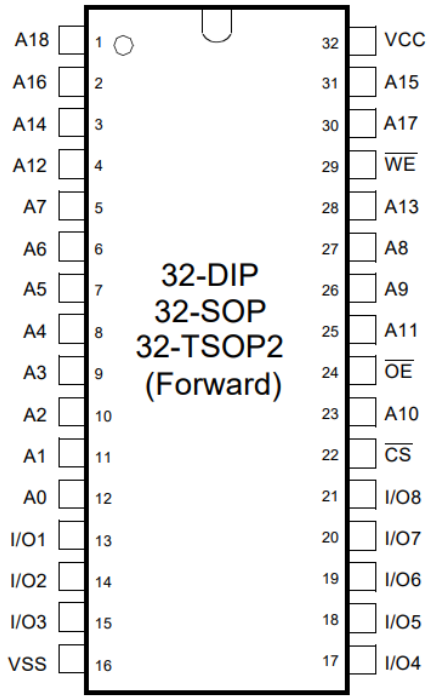


Рисунок 5.15 – Зображення функціональної схеми K6T4008C1B-GB55

Memory array (Масив пам'яті): Основний блок пам'яті складається з 1024 рядків та 512 x 8 колонок, що дозволяє зберігати 8-бітні дані на кожен осередок пам'яті. Рядки і стовпці організовані таким чином, щоб забезпечити ефективний доступ до даних через адреси, що вказують на конкретні осередки пам'яті.

Row select (Схема вибору рядка): обирає конкретний рядок пам'яті для доступу, коли надано адресу рядка через вхідні лінії A0-A18. Вибір конкретного рядка є першим етапом для доступу до даних пам'яті, після чого можна перейти до вибору стовпців.

I/O Circuit (I/O схема): займається обробкою введення-виведення даних з пам'яті. Вона організовує підключення між внутрішньою пам'яттю та зовнішнім інтерфейсом, що забезпечує передачу даних між процесором або іншими пристроями і осередками пам'яті.

Column select (Схема вибору стовпця): Після того, як вибрано рядок, схема вибору стовпця відповідає за вибір конкретного стовпця пам'яті для доступу до даних. Адреса стовпця формується через лінії адреси A2-A9, A10-A17, що дозволяє вибрати точний осередок для зчитування або запису.

Data cont (Контролер даних): управляє передачею інформації між пам'яттю і зовнішнім обладнанням. Він координує процеси зчитування і запису даних до/з пам'яті, контролюючи як вхідні, так і вихідні лінії.

Control logic (Контрольна логіка): Цей блок обробляє сигнали керування, такі як CS (Chip Select), WE (Write Enable) та OE (Output Enable), що відповідають за визначення, чи є операція читання чи запису, а також за активування відповідних ліній доступу до пам'яті.

I/O1...I/O8 (Вхідні/вихідні лінії): Це група ліній, через які дані передаються між зовнішнім обладнанням і внутрішнім масивом пам'яті. Вхідні сигнали на цих лініях використовуються для запису даних у пам'ять, а вихідні — для зчитування.

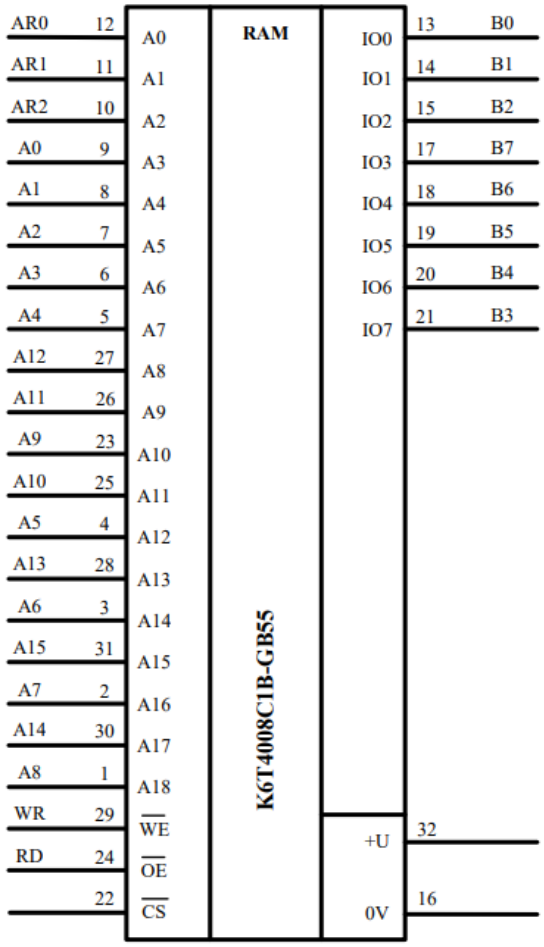


Рисунок 5.16– Зображення принципіальної схеми K6T4008C1B-GB55

Таблиця 5.4 - Входи/виходи K6T4008C1B-GB55

Позначення	Номер виводу	Опис
WE	29	Вхід дозволу запису
CS	22	Вхід вибору мікросхеми
OE	24	Вхід дозволу виводу
A0-A18	12-30	Адресні входи
I/O1-I/O8	13-21	Входи/виходи даних
VCC	32	Живлення
VSS	16	Заземлення

Часові діаграми K6T4008C1B-GB55:

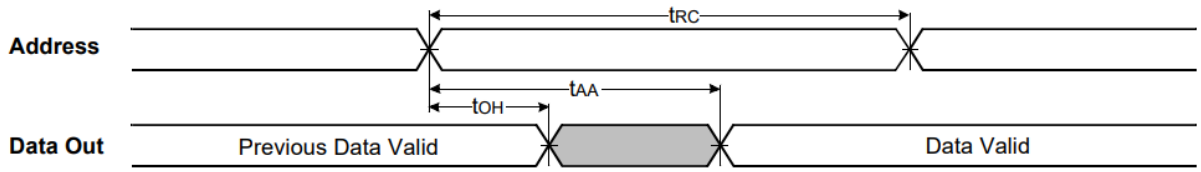


Рисунок 5.17 – Форма сигналу синхронізації циклу зчитування

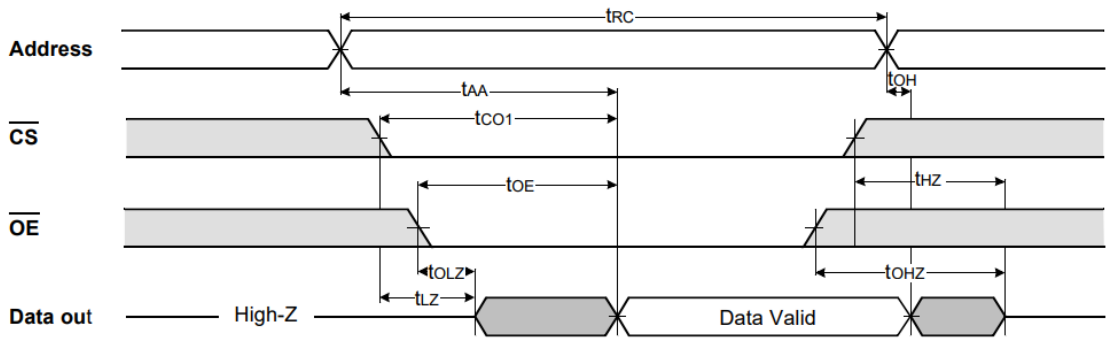


Рисунок 5.18 - Форма хвилі синхронізації циклу зчитування

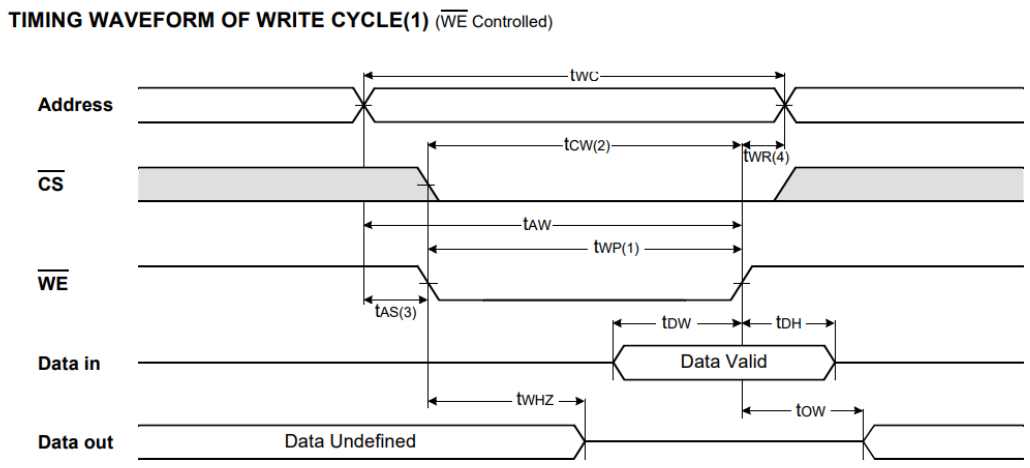


Рисунок 5.19 – Часова форма хвилі циклу запису

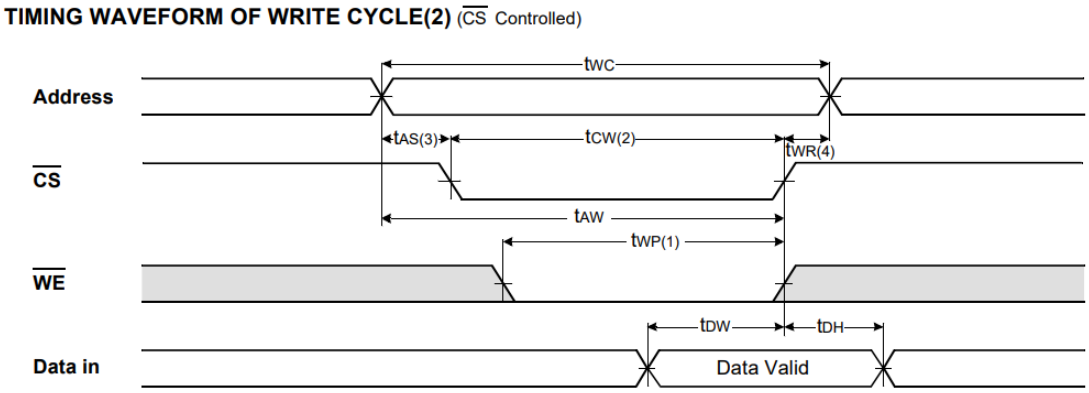


Рисунок 5.20- Форма сигналу синхронізації циклу запису

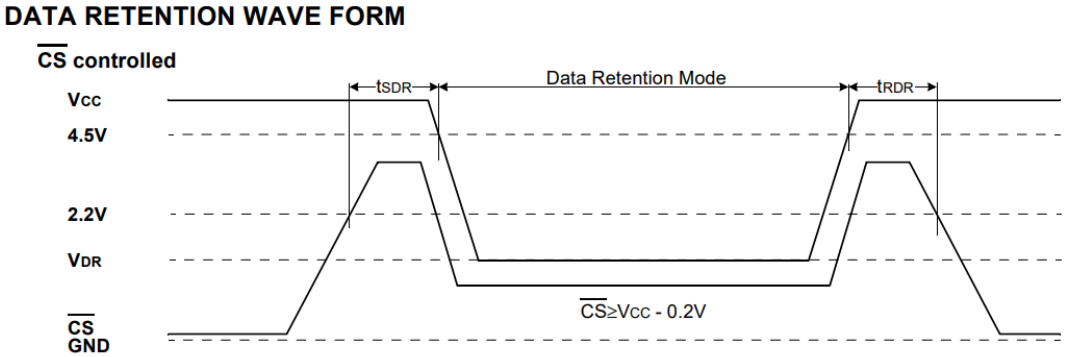


Рисунок 5.21 - Форма хвилі збереження даних

Характеристика K6T4008C1B-GB55:

Напруга живлення: +5V (діапазон від 4,5 V до 5,5 V).

Сигнали для логічних рівнів: працюють на логічних рівнях 0 (низький) і 1 (високий), з рівнем напруги, що відповідає логічним стандартам для 5V систем.

Потужність споживання залежить від режиму роботи мікросхеми та умов навантаження, але зазвичай споживана потужність складає приблизно 40 мВт в стандартних умовах. У режимі очікування (standby) споживана потужність може знижуватись.

Рівень напруги сигналів мікросхеми:

•Логічний рівень "0" (LOW): 0 - 0.8 V.

•Логічний рівень "1" (HIGH): 2.4 - 5 V (для 5V системи).

•Ці рівні відповідають стандарту TTL/CMOS.

Коефіцієнт розгалуження вихідних сигналів залежить від конкретного режиму роботи мікросхеми та її навантаження. Оскільки це пам'ять з синхронним доступом, виведення даних на лінії I/O зазвичай потребує підключення безпосередньо до інших логічних схем або процесорів, де вони передаються за допомогою шин або послідовних інтерфейсів.

Обсяг пам'яті - мікросхема має 1024 рядки по 8 біт (512 × 8), що дає 8 кілобайт пам'яті.

Типи інтерфейсів:

•I/O інтерфейси: Лінії вводу/виводу даних (I/O1-I/O8), через які здійснюється обмін даними з зовнішнім пристроєм.

•Адресний інтерфейс: Лінії A0-A18, які використовуються для адресації рядків та стовпців пам'яті.

•Інтерфейс управління: Включає лінії CS (Chip Select), WE (Write Enable), і OE (Output Enable), що дозволяють контролювати операції з пам'яттю.

5.5 Розробка програмного забезпечення пристрою

Програма нижче реалізує алгоритм шифрування:

AVR GCC – компілятор для мов C і C++ під мікроконтролери AVR.
Avr-libs – стандартна бібліотека мови C для розробки з GCC.
Avr-as – асемблер для AVR-мікроконтролерів.
#include <stdio.h>

```
#include <stdlib.h>
#include <string.h>

#define MAX_LENGTH 1000

void encryptMessage(char *message, int shift, char *keyword) {
    int messageLength = strlen(message);
    int keywordLength = strlen(keyword);
    int i, j;

    for (i = 0, j = 0; i < messageLength; i++) {
        char currentChar = message[i];
        int charShift = shift;

        if (j < keywordLength) {
            charShift += keyword[j];
            j++;
        }

        if (currentChar >= 'a' && currentChar <= 'z') {
            currentChar = 'a' + (currentChar - 'a' + charShift) % 26;
        } else if (currentChar >= 'A' && currentChar <= 'Z') {
            currentChar = 'A' + (currentChar - 'A' + charShift) % 26;
        }

        message[i] = currentChar;
    }
}

int main() {
    char message[MAX_LENGTH];
    int shift;
    char keyword[MAX_LENGTH];
    char confirm;

    printf("Введіть текст для шифрування: ");
    fgets(message, sizeof(message), stdin);
    message[strcspn(message, "\n")] = '\0';

    printf("Підтвердіть продовження (Y/N): ");
    scanf(" %c", &confirm);

    if (confirm != 'Y' && confirm != 'y') {
        printf("Програму завершено.\n");
```

```
        return 0;
    }

    printf("Оберіть мову (1 - українська, 2 - англійська): ");
    int language;
    scanf("%d", &language);

    printf("Введіть числовий ключ (K): ");
    scanf("%d", &shift);

    printf("Введіть ключове слово: ");
    scanf("%s", keyword);

    encryptMessage(message, shift, keyword);

    printf("Зашифроване повідомлення: %s\n", message);

    return 0;
}
```

6 ТЕХНІКО-ЕКОНОМІЧНА ЧАСТИНА

6.1 Розрахунок повної собівартості проектованого пристрою

Собівартість продукції є одним із найважливіших показників для підприємства, оскільки вона визначає ефективність виробництва та дозволяє встановити ціну товару на ринку. Калькулювання собівартості — це процес визначення всіх витрат, пов’язаних з виробництвом і реалізацією продукції, що дозволяє підприємству оцінити, які ресурси витрачаються на виготовлення кожної одиниці продукції. Собівартість може включати витрати на сировину, матеріали, працю, утримання обладнання, а також інші накладні витрати, і є важливим інструментом для встановлення економічної доцільності виробництва. Важливою складовою цього процесу є систематизація витрат і правильна класифікація за відповідними статтями.

Собівартість продукції— це сукупність витрат, які йдуть для виготовлення продукції та її подальшої реалізації. Ці витрати виражаються у грошовому еквіваленті та дозволяють визначити, скільки коштує виробництво кожної одиниці продукції. Витрати на виробництво складають виробничу собівартість, а разом із витратами на збут формують повну собівартість продукції. Розрахунок собівартості конкретного виробу по статтях витрат називається калькуляцією.

Калькуляція собівартості є важливим етапом планування та обліку, оскільки дозволяє визначити точний рівень витрат на виробництво і реалізацію продукції. Важливою особливістю калькуляції є те, що вона допомагає не лише розрахувати собівартість, але й виявити, на яких етапах виробництва чи в якій складовій виникають найбільші витрати. Це допомагає у подальшому приймати обґрунтовані управлінські рішення для зменшення витрат і підвищення ефективності виробництва.

Одним із основних етапів у калькулюванні собівартості є класифікація витрат за окремими статтями, які детально розкривають усі складові витрат підприємства. Вони поділяються на кілька груп залежно від їхнього характеру і призначення. Стандартно в обліку собівартості продукції застосовують таке групування витрат за статтями калькуляції:

– Основна заробітна плата. Це сума, що виплачується працівникам безпосередньо за виконання виробничих функцій, що безпосередньо

пов’язані з виготовленням продукції. Включає основну заробітну плату робітників і спеціалістів, які безпосередньо працюють над виробничим процесом.

- Додаткова заробітна плата. Це сума доплат та премій, які виплачуються працівникам за досягнення певних результатів, перевиконання плану чи виконання додаткових обов’язків. Це може бути виплата бонусів або компенсацій.
- Відрахування від заробітної плати. Це внески, які утримуються з заробітної плати працівників, такі як соціальні внески, пенсійні відрахування, страхові внески тощо.
- Матеріали та комплектуючі. Це витрати на сировину, матеріали, комплектуючі та інші ресурси, які використовуються в процесі виробництва продукції. Сюди входять як основні матеріали, так і допоміжні (фарби, хімічні речовини тощо).
- Витрати на утримання та експлуатацію обладнання. Це витрати на обслуговування та експлуатацію машин, інструментів і інших засобів виробництва. Вони включають вартість енергоносіїв, ремонти, технічне обслуговування, амортизацію та інші витрати, що стосуються підтримки виробничого обладнання.
- Виробничі витрати. Це різноманітні витрати, які не входять до інших категорій, але безпосередньо пов’язані з процесом виробництва. До них належать витрати на утримання виробничих приміщень, комунальні послуги, витрати на охорону праці тощо.
- Адміністративні витрати. Це витрати, що пов’язані з управлінням підприємством, включаючи оплату праці адміністративного персоналу, витрати на офіс, канцелярські товари, адміністративне забезпечення тощо.
- Позавиробничі витрати (комерційні витрати). Це витрати, що виникають у процесі реалізації продукції на ринку, такі як витрати на рекламу, транспортування, зберігання товару, організацію збуту тощо.

Ці статті калькуляції є основою для розрахунку собівартості продукції і допомагають детально відобразити всі етапи витрат, що виникають в процесі її виробництва та продажу.

6.1.1 Матеріали та комплектуючі

Матеріали та комплектуючі аналізуються на основі інформації, яка надається в різних джерелах, таких як каталоги, прайс-листи, веб-сайти виробників і постачальників, а також інші ресурси, що містять дані про матеріали, сировину та комплектуючі. Ця інформація враховує витрати на операції, розраховані на одну одиницю продукції.

Дані витрат наведені у таблиці 6.1.

Таблиця 6.1 – Розрахунок витрат на комплектуючі

№ п/п	Найменування	Кількість, од.	Ціна за одиницю, грн.	Сума, грн.
1	2	3	4	5
Мікросхеми				
1	74HCT573	2	20	40
2	K6T4008C1B-GB55	1	100	100
3	MAX232EPE	1	63	63
4	ATMEGA8535	1	140	140
Конденсатори				
1	1000uF 10V	11	7	77
2	10uF 50V 5x11mm HY	5	1,30	6,5
3	0,1мкФ 1000В ±5% CBB20	5	11	55
Резистори				
1	CRL-W 10kOm	6	10	60
Резонатори				
1	6.0MHz HC-49S	1	6	6
Роз’єми				
1	*DB-9*	3	13	39
Разом				586,5

Сума за всі комплектуючі становить 586,5 гривень

Розрахунок витрат за матеріали наведений у таблиці 6.2.

Таблиця 6.2 – Розрахунок витрат за матеріали

Матеріал	Одиниця вимірювання	Норм витрат	Ціна за од, грн.	Ціна, грн.
1	2	3	4	5
Провід монтажний	м	0,3	3	0,9
Склотекстоліт	$м^2$	0,2	75	15
Каніфоль	кг	0,1	1050	105
Флюс	кг	0,03	910	27,3
Припій	кг	0,2	340	68
Лак	кг	0,04	110	4.4
Речовина для корпусу	кг	0,3	280	84
Разом, М				304,6

З урахуванням транспортно-заготівельних витрат ($k_{т-з}$=5 – 15%) вартість комплектуючих та матеріалів становитиме:

$$КМ = (К+М) * (100+k_{т-з}) / 100, \quad (6.1)$$

$$КМ = (586,5+304,6) * (100+10) / 100= 980,21 \text{ грн.}$$

6.1.2 Витрати на основну заробітну плату

$$Зо = \sum_{i=1}^{n} Тг_i \cdot Нч_i , \quad (6.2)$$

де Тг - це годинна зарплата працівника (наприклад, інженера, лаборанта), який бере участь у виробництві пристрою, грн/год.

Нчі – це час, який робітник витрачає на створення і налаштування пристрою, год, Нчі=4год.

n - число працівників, які безпосередньо беруть участь у виготовленні пристрою. n=2.

Заробіток за годину роботи визначається на основі місячної зарплати працівника:

$$Тг_i = \frac{Тм_i}{Вф_i \cdot 8}, \quad (6.3)$$

де $Тм_i$ – місячна зарплата працівника, грн.

$Вф_i$ - кількість днів або змін, які працівник фактично відпрацював за місяць.

8 – кількість годин які відпрацьовує працівник.

$$Тг_i = \frac{Тм_i}{Вф_i \cdot 8} = \frac{14000}{12 \cdot 8} = 145 \text{ грн.}$$

$$З_о = \sum_{i=1}^{2} 145 \cdot 5 = 2 \cdot 145 \cdot 5 = 1450 \text{ грн}$$

6.1.3 Витрати на додаткову заробітню плату

Додаткова заробітна плата (10-30% від Зо):

$$Зд = Зо \cdot \frac{Кд}{100}, \quad (6.4)$$

Кд - це додаткова оплата, яку працівник отримує, Кд = 10%.

$$Зд = 1450 \cdot \frac{10\%}{100\%} = 145 \ (грн)$$

6.1.4 Відрахування на соціальні виплати

Відрахування на соціальні виплати охоплюють внески, розраховані на основі основної та додаткової заробітної плати за чинними тарифами. До них належать:

- обов’язкові внески до пенсійного фонду;
- страхові платежі у випадках нещасних випадків на виробництві;
- обов’язкові внески до фонду соціального страхування від безробіття;
- витрати, пов’язані з тимчасовою непрацездатністю;
- витрати, пов’язані з виплатами при народженні дитини та організацією поховання;

Нарахування на заробітну плату – єдиний соціальний внесок у розмірі 22%.

$$В_{св}=(З_о+З_д)*22/100, \quad (6.5)$$

$$В_{св}=(1450+145)*22/100=350,9 \text{ (грн)}$$

6.1.5 Видатки на утримання та експлуатацію встаткування

Витрати на утримання та експлуатацію встаткування (ВУЕ) перебувають на балансі підприємства міста і розраховуються за такою формулою:

ВУЕ = основна заробітна плата * відсоток ВУЕ (приймають відсоток ВУЕ рівним 120 ÷ 150%).

$$ВУЕ=1450*1,2=1740 \text{ грн}$$

6.1.6 Загальновиробничі витрати

Загальновиробничі витрати включають різні види витрат, пов'язані з управлінням підрозділом. Сюди належать витрати на організацію управлінських процесів у межах підрозділу, службові відрядження працівників, амортизація основних засобів загальноцехового призначення та інші аналогічні витрати.

Загальновиробничі витрати (В зв) визначаються у розмірі 130-250% від основної заробітної плати.

$$В_{зв}=З_о*\%В_{зв}=1450*1,3=1885 \text{ (грн)}, \quad (6.6)$$

6.1.7 Виробнича собівартість

Виробнича собівартість розраховується за формулою

$$В_с = З_о + З_д + В_{св} + КМ + ВУЕ + В_{зв}=$$
$$=1450 + 145 + 350,9 + 980,21 + 1740 + 1885 = 6551,11, \quad (6.7)$$

6.1.8 Адміністративні витрати

Адміністративні витрати можуть охоплювати такі складові:

– витрати на управління підприємством, включаючи планування, координацію та контроль за діяльністю;
– витрати на службові відрядження адміністративного персоналу;

- витрати на утримання пожежної та сторожової охорони на підприємстві;
- витрати на навчання та перепідготовку працівників для підвищення їхньої кваліфікації;
- витрати на організацію транспортування працівників до місця роботи і назад;
- витрати на сплату відсотків за фінансові та комерційні кредити;
- витрати, пов'язані з орендою чи лізингом матеріальних цінностей;
- витрати на оплату послуг банків і фінансових установ;
- податки та інші обов'язкові платежі.

Адміністративні витрати ($В_а$) визначаються у розмірі 140 - 200% від основної заробітної плати.

$$В_а = З_о * В_а = 1450 * 1,4 = 2030 \text{ (грн)}, \quad (6.8)$$

6.1.9 Витрати на збут

Витрати на збут ($В_з$) включають різні компоненти, зокрема витрати на рекламу та передреалізаційну підготовку пристрою. Зазвичай вони становлять 5–10% від виробничої собівартості і є ключовим елементом для забезпечення ринкової активності та успішного виведення пристрою на ринок.

Рекламні витрати охоплюють витрати на організацію рекламних кампаній, створення промоційних матеріалів, проведення заходів із просування та інші дії, спрямовані на підвищення видимості продукту.

Передреалізаційна підготовка пристрою включає витрати на тестування, сертифікацію, оформлення документації, ліцензування та інші дії, необхідні для введення пристрою в експлуатацію.

Ці витрати сприяють формуванню підтримки на ринку, підвищенню проінформованості клієнтів та стимулюванню попиту на товар.

$$В_з = В_с * (5\text{-}10\%) = 6551,11 * 0,05 = 327,55 \text{ (грн)}, \quad (6.9)$$

6.1.10 Повна собівартість пристрою

Повна собівартість пристрою (С) розраховується за формулою:

$$С = В_с + В_а + В_з, \quad (6.10)$$

$$С = 6551,11 + 2030 + 327,55 = 8938,66 \text{ (грн.)}$$

Калькуляцію собівартості виробу зведено в таблицю 6.3.

Таблиця 6.3 – Калькуляція собівартості пристрою

№	Найменування статей калькуляції	Значення, грн.
1.	Основна заробітна плата	1450
2.	Додаткова заробітна плата	145
3.	Відрахування на соціальні виплати	350,9
4.	Видатки на утримання та експлуатацію встаткування	1740
5.	Загальновиробничі збори	1885
6.	Матеріали та комлектуючі	980,21
Виробнича собівартість		6551,11
7.	Адміністративні витрати	2030
8.	Витрати на збут	327,55
Повна собівартість пристрою		8908,66

6.2 Розрахунок ціни пристрою

6.2.1 Розрахунок оптової ціни пристрою

У ринковій економіці використовуються різні підходи до формування цін, зокрема методи, що базуються на собівартості з додаванням прибутку, забезпеченні заданого рівня доходу чи врахуванні рівня попиту.

Для визначення оптової ціни пристрою скористаємося підходом «собівартість плюс прибуток»

$$Ц_{опт} = С + П, \qquad (6.11)$$

Де С – собівартість пристрою.

П – величина прибутку.

Розмір прибутку розраховується на основі нормативного рівня рентабельності виробництва продукції:

$$R = (П / С) * 100\%, \quad (6.12)$$

Де R – рентабельність продукту, що приймається у розмірі до 35%.

R = 10%

Тоді оптова ціна буде:

$$Ц_{опт} = С+(R*С/100)= 8908,66 + (10 * 8908,66 / 100) = 9800 \text{ (грн.)} \quad (6.13)$$

6.2.2 Розрахунок роздрібної ціни пристрою

Визначимо роздрібну ціну розробленого пристрою при 20% ПДВ.

$$Ц_{розн} = Ц_{опт} * 1,2 = 9800 * 1,2 = 11760 \text{ (грн.)}, \quad (6.14)$$

Методика визначення ціни, описана вище, має як переваги, так і недоліки. До її сильних сторін належать простота та зрозумілість, особливо в аспекті відшкодування витрат на виробництво й забезпечення прибутковості від створення та реалізації пристрою.

Проте ця методика має обмеження, оскільки недостатньо враховує ринкові фактори, такі як попит, що може призвести до невідповідності встановленої ціни реальним ринковим умовам. Особливо це стосується недооцінки впливу конкуренції, державного регулювання рентабельності продукції та інших чинників ринку.

Таким чином, ця методика є доцільною лише в окремих ситуаціях, наприклад, за умов монополії, обмеженої рентабельності, одноразових замовлень або виробництва унікальної продукції.

Для встановлення адекватної ціни, яка відповідає сучасним ринковим реаліям, необхідно проводити додаткові маркетингові дослідження, що враховують конкуренцію, попит та інші ключові ринкові фактори.

ВИСНОВКИ

У результаті виконання магістерської кваліфікаційної роботи була створена електронна система захисту інформації на базі моноалфавітного алгоритму шифрування, яка повністю відповідає технічним вимогам завдання.

У процесі роботи була досліджена криптостійскість моноалфавітних шифрів, варіативність моноалфавітних шифрів та розкриті сильні та слабкі їх сторони.

Розроблено алгоритм функціонування електричної системи, структурну схему, електричну принципову схему, електричну функціональну схему та програмне забезпечення до електронної системи.

При проведенню економічного аналізу проведено розрахунок собівартості системи, який показав його рентабельність. Таким чином, створення системи виявилося обґрунтованим як з технічної, так і з економічної точки зору.

Розроблена система має потенціал для подальшого вдосконалення та масштабування, що відкриває можливості її застосування в різних сферах, таких як захист конфіденційної інформації, автоматизовані системи обміну даними чи інтеграція у комплексні системи кібербезпеки.

СПИСОК ЛІТЕРАТУРИ

1. Щур Н.О., Покотило О.А. Основи криптології: навч. посібник. – Житомир: Державний університет «Житомирська політехніка», 2021. 120 с.
2. Шифрування з симетричними ключами [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Шифрування_з_симетричними_ключами
3. Асиметричні алгоритми шифрування [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Асиметричні_алгоритми_шифрування
4. Частотний аналіз [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Частотний_аналіз
5. Моноалфавітні шифри [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Моноалфавітні_шифри
6. Поліалфавітний шифр [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Поліалфавітний_шифр
7. Шифр Атбаш [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Атбаш
8. Шифр Прейфера [Електронний ресурс]: - https://uk.wikipedia.org/wiki/Шифр_Плейфера
9. Alfred J. Menezes. Handbook of Applied Cryptography/ Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. Publisher: CRC Press, 2001. – 780 pages
10. Jean-Philippe Aumasson. Serious Cryptography: A Practical Introduction to Modern Encryption Paperback. Kindle Edition, 2017. – 313 pages.
11. Козіна Г.Л. Криптографія від історії до сучасних стандартів: навч.посібник / Г. Л. Козіна. – Запоріжжя : НУ «Запорізька політехніка», 2020. – 192 с
12. Юрченко В.І., Бортова система вагового контролю автомобіля / Бережна О.В., Горячев О.Є., Юрченко В.І., Мельник Р.В., Мороз М.В. // Фізика, електроніка, електротехніка (ФЕЕ-2023). Матеріали та програма науково-технічної конференції. – Суми: СумДу, 2023. – С.72.

Бортова система вагового контролю автомобіля

Бережна О.В., *доцент*; Горячев О.Є., *ст. викладач*;
Юрченко В.І., *студент гр. ЕС.м-21*;
Мельник Р.В., *студент гр. ЕС-91*; Мороз М.В., *студент гр. ЕС.м-21*
Сумський державний університет, м. Суми, Україна

Для підвищення ефективності здійснення вантажоперевезень, для підвищення рівня безпеки на дорогах та для мінімізації надмірного зносу дорожнього полотна під впливом вантажівок, що порушують встановлені для них вагогабаритні норми, необхідно впровадження пристроїв та систем контролю навантаження на вісь автомобілів та оцінки ваги вантажу, що перевозиться.

Серед сучасних систем вагового контролю автомобілів виділяється три типи бортових систем зважування, таких як гідравлічна, пневматична та механічна. Системи, які базуються на вимірюванні тиску мастила в гідравлічній системі або повітря в пневматичній, мають загальні недоліки – складність налаштування системи та низька ремонтопридатність. Перспективним для створення бортових систем зважування бачиться універсальне рішення для багатьох типів автомобілів з використання тензодатчиків та їх встановлення на кожну вісь вантажівки.

Аналіз показав, що таке рішення забезпечує незалежність від типу підвіски автомобіля, високу надійність, великий період експлуатації, високу точність, відсутність похибки від зовнішніх умов та стану підвіски. Для зменшення вартості такого рішення пропонується використовувати оптимальну кількість тензодатчиків, одного мікроконтролеру з необхідним програмним забезпеченням замість блоку аналізу і розподільної коробки та під'єднання тензодатчиків до мікроконтролеру за допомогою модуля підсилення сигналу, що спрощує експлуатацію, збільшує модифікаційні можливості системи та полегшує ремонт.

Системи вагового контролю, що базуються на тензодатчиках, можуть здійснювати контроль навантаження, що припадає на кожну з осей автомобіля, визначати вагу вантажу, що перевозиться автомобілем, сигналізувати про перевищення осьового навантаження. Параметри, що визначаються, можна відображати на дисплеї водія і передавати до відповідних систем контролю та моніторингу.