НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРІЇНИ «КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

(повна назва інституту/факультету)

Кафедра електронних приладів та пристроїв

(повна назва кафедри)

«На правах рукопису»
УДК 004.056.55(004.421.5)

«До захисту допущено»
Завідувач кафедри
_____ Писаренко Л.Д.
(підпис) (ініціали, прізвище)
"____"_____20____р.

МАГІСТЕРСЬКА ДИСЕРТАЦІЯ

Зі спеціальності (спеціалізації) 171- електронні прилади та пристрої

На тему: Апаратно-програмна модифікація криптографічних методів захисту інформації у ПК

Виконав (-ла): студент (-ка) 2 курсу, групи ДЕ-81мп
Іванов Роман Євгенович _____
(прізвищу, ім'я, по батькові) (підпис)

Науковий керівник: Писаренко Л.Д. _____
(прізвищу, ім'я, по батькові) (підпис)

Консультант: _____ _____
(прізвищу, ім'я, по батькові) (підпис)

Рецинзент: _____ _____
(прізвищу, ім'я, по батькові) (підпис)

Нормконтроль: Чадюк В.О _____
(прізвищу, ім'я, по батькові) (підпис)

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____
(підпис)

Київ – 2019 року

Національний технічний університет України
«Київський політехнічний інститут
імені Ігоря Сікорського»

Інститут/факультет: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського»
(повна назва)

Кафедра: Електронних приладів та пристроїв
(повна назва)

Рівень вищої освіти – другий (магістерський) за освітньо-професійною (освітньо-науковою) програмою

Спеціальність (спеціалізація) 171 – електронні прилади та пристрої
(код і назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____ _____
(підпис) (ініціали, прізвище)
«___»_____20__ р.

		Іванов Р.Є.			МРП.171.081.004	Лист
		Писаренко Л.Д.				
Вим.	Арк	№ докум.	Підпис	Дата		

ЗАВДАННЯ

на магістерську дисертацію студенту

Іванов Роман Євгенович

(прізвище, ім'я, по батькові)

1. Тема дисертації: Апаратно-програмна модифікація криптографічних методів захисту інформації у персональних комп`ютерах

науковий керівник дисертації: професор, д.т.н., Писаренко Леонід Дмитрович

(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від «07» 11 2019 р. № 3859-с

2. Строк подання студентом дисертації: 13.12.2019

3. Об'єкт дослідження: Криптографічний метод захисту інформації у вигляді генератора псевдовипадкових послідовностей.

4. Предмет дослідження (Вихідні дані – для магістерської дисертації за освітньо-професійною програмою): Огляд генераторів псевдовипадкових послідовностей, розробка структурної та принципової схеми пристрою, розробка та дослідження складових частин приладу.

5. Перелік завдань, які потрібно розробити: Графічні ілюстрації структурної та електрично принципової схем пристрою(що розробляється), розробка та дослідження складових частин пристрою (що розробляється).

6. Перелік графічного (ілюстративного) матеріалу: Графічна ілюстрація структурної схеми приладу, графічна ілюстрація електрично принципової схеми, ілюстраційний плакат з графіками досліджень.

7. Орієнтовний перелік публікацій:

1. Наукова конференція факультету електроніки – 2017 рік.
2. Наукова конференція факультету електроніки – 2019 рік.

8. Дата видачі завдання 01.09.2018

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Строк виконання етапів магістерської дисертації	Примітка
1	Огляд науково-технічної літератури по генераторам псевдовипадкових послідовностях та квантовій криптографії	08.10.2019	
2	Розробка структурної та принципової схем, розробка та дослідження фізико-математичної моделі (пристрою, що розробляється).	25.10.2019	
3	Вибір складових частин пристрою та визначення їх функцій, створення модифікованої програми.	03.11.2019	
4	Оформлення графічної частини, пояснювальної записки, плакатів, підготовка доповіді.	17.11.2019	

Студент _____ Іванов Р.Є._____

(підпис) (ініціали, прізвище)

Науковий керівник дисертації _____ Писаренко Л.Д.

(підпис) (ініціали, прізвище)

З М І С Т

Р Е Ф Е Р А Т

«Апаратно-програмна модифікація криптографічних методів захисту інформації у персональних комп`ютерах»

Магістерська робота напряму підготовки **171 - «Електронні прилади та пристрої». Іванов Роман Євгенович.** Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського. Факультет електроніки, кафедра електронних приладів та пристроїв. Група ДЕ-81мп. – К.: Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського» 2019. – 56 с., іл. 8, табл. 13.

Ключові слова: Криптосистема, генератор псевдовипадкових послідовностей, ассиметричні та симетричні криптоалгоритми, квантова криптографія, шифрування, тривалість затримки, практична та абсолютна стійкість криптоалгоритмів, шифрування з ключем,електронно-цифровий підпис.

Короткий зміст роботи: Дана дипломна робота присвячена розгляду методів покращення криптографічного захисту інформації, в ній представлено модифікації алгоритмів та їх реалізація.

Вступ описує актуальність теми та задачу роботи. В літературі оглянуто генератори псевдовипадкових послідовностей, розглянута квантова криптографія та приклад передачі інформації за допомогою квантової криптографії. В теоретичній частині приведені основні методи захисту інформації та критерії якості псевдовипадкових послідовностей. В конструкторсько-технологічній частині приведена робота пристрою та його складові частини. В експериментальній частині розроблена структурна та принципова схема.

А Н О Т А Ц І Я

Дана дипломна робота присвячена вивченню методів захисту інформації за допомогою криптографії , в ній виконано вдосконалення одного із алгоритмів захисту та створення пристрою,відповідно даній модифікації , дослідженням його складових частин та фунцій, які вони виконують.

Робота складається із вступу, аналізу літератури, теоретичної частини, методики інженерних розрахунків та висновків. Вступ описує актуальність теми та задачу роботи. В літературі оглянуто генератори псевдовипадкових послідовностей, розглянута квантова криптографія та приклад передачі інформації за допомогою квантової криптографії. В теоретичній частині приведені основні методи захисту інформації та критерії якості псевдовипадкових послідовностей. В конструкторсько-технологічній частині приведена робота пристрою та його складові частини. В експериментальній частині розроблена структурна та принципова схема.

S U M M A R Y

This diploma project is devoted to the research of methods for improving cryptographic protection of information, it’s presented the modifications of protection algorithms and device modeling, which will correspond to these modifications, research of its constituent parts and functions, which they perform.

The work consists of introduction, analysis of literature, theoretical part, methods of engineering calculations and conclusions. The introduction formulated the main task of the work and shows its relevance. In the literature analysis, a review of generators of pseudorandom sequences is presented, quantum cryptography is considered and an example of information transfer with quantum cryptography is considered. The theoretical part contains the basic methods of information protection and quality criteria for pseudorandom sequences. In the design and technological part the operation of the device and its components are given. In the experimental part a structural and principal scheme was developed.

Перелік умовних позначень, символів та скорочень

ГПСП- генератори псевдовипадкових послідовностей

ГВЧ- генератори випадкових чисел

ЕЦП-електронно-цифровий підпис

Вступ

Проблема захисту інформації - одна з найбільш актуальних інженерно-технічних проблем сучасності. В останні роки вона почала привертати увагу не тільки криптографів, але і технічних фахівців широкого профілю та широкої громадськості. Пов'язано це, в першу чергу, з безпрецедентними темпами розвитку комп'ютерних мереж, які за останні 10-15 років перетворилися із засобу передачі більш-менш публічної інформації в засіб забезпечення економічної діяльності організацій і окремих громадян. Електронна комерція, електронні перекази грошових коштів і тому подібні нові прикладні застосування комп'ютерних мереж, очевидно, відрізняються від більш традиційних, пов'язаних з передачею відкритої інформації або особистого листування, на порядки більшою цінністю інформації в грошовому еквіваленті і відповідною величиною збитків від її розголошення. При незмінній технічній складовій мереж це призводить до радикальної зміни співвідношення цінності інформації до вартості отримання несанкціонованого доступу до неї, що може привести - і призводить на практиці - до активізації діяльності по отриманню такого доступу. Перешкоджати цьому, очевидно, життєво необхідно з економічної точки зору для будь-якої організації, яка здійснює діяльність у одній з названих областей, дохід якої залежить безпосередньо від надійності передачі даних контрагенту або репутації якої може бути завдано істотної шкоди внаслідок шкоди, завданої клієнту несанкціонованим доступом третьої особи до конфіденційної інформації в процесі обслуговування.

Якщо позначену вище проблему можна назвати пов'язаної і інтенсивним розвитком комп'ютерних мереж, то існує і ряд проблем,

пов'язаних з їх екстенсивним розвитком: збільшенням кількості пристроїв в мережах, більш широким географічним охопленням і поширенням серед населення. Наслідком цього, зокрема, може бути проходження внутрішньої

інформації деякої організації через апаратуру, що належить іншій організації, наприклад, передача внутрішньокорпоративної пошти між відділеннями в різних країнах, через магістральну лінію передачі даних, непідконтрольну організації. З іншого боку, поширення серед населення коштів доступу до комп'ютерних мереж неминуче призводить до зниження середнього рівня технічної грамотності користувачів. Ці обставини переконливо демонструють, що,

Третьою складовою проблеми є практична неможливість переходу на повсюдне використання приладів, які володіють високою криптостійкістю обчислювально-інтенсивних методів шифрування даних. Однією причиною цього є інерційність величезної маси учасників інформаційного обміну, оскільки такий перехід спричинив би за собою необхідність переробки надзвичайно великих обсягів матзабезпечення. Інша причина - в широкому поширенні мобільних пристроїв, часто які не мають достатніх обчислювальних ресурсів, але, тим не менш, використовуваних в інформаційному обміні і тим більше потребують захисту, через те, що передача даних мобільними пристроями часто відбувається бездротовим чином, по радіоканалу, схильній до перехоплення зловмисником.

Метою написання дипломної роботи є розгляд методів покращення криптографічного захисту інформації.

1. Огляд науково-технічної літератури

1.1 Огляд генераторів псевдовипадкових послідовностей

Детерміновані ГПСП

Це арифметичний генератор,який має велику швидкість, але недоліків у нбого набагато більше:оборотність, не достатньо довгі періоди,залежність деяких послідовних значень.

Серед цих генераторів більшою популярнісю користуються метод Фібоначчірегістр зсуву з лінійним зворотним зв'язком,регістр зсуву з узагальненим зворотним зв'язком.

В теперішніх ГПСП популярний «вихор Мерсенна», винайдений в 1997 році Мацумото і Нісимура. Його перевагами є колосальний період, рівномірне розподілення в 623 вимірах,швидкісне генерування випадкових чисел (в декілька разів скоріше, ніж стандартні ГПСП, використовують лінійний конгруентний метод). Однак, існують алгоритми, що розпізнають послідовність, що породжуються вихором Мерсенна, як невипадково.

Апаратні ГПСП

Апаратний генератор випадкових чисел - пристрій, що генерує послідовності випадкових чисел на основі вимірюваних параметрів протікаючого фізичного процесу. Робота таких пристроїв часто заснована на процесах рівня елементарних частинок, таких як тепловий шум, фотоелектричний ефект, інші квантові явища. Ці процеси, в теорії, абсолютно непередбачувані. Апаратні генератори випадкових чисел, засновані на квантових процесах, зазвичай складаються зі спеціального підсилювача і перетворювача. Підсилювач підсилює дуже слабкі сигнали,

отримані в результаті протікання фізичних явищ, до прийнятних розмірів, які перетворюються перетворювачем до цифрової форми .

Апаратні генератори випадкових чисел дещо повільні і можуть виробляти зміщені послідовності (коли певна послідовність чисел повторюється частіше). Використання подібних генераторів залежить від потреб конкретної предметної області і від пристрою самого генератора.

Майже всі існуючі комерційні апаратні ГПСП запатентовані або тримаються в секреті. Апаратні ГПСП обмежені строгими вимогами до витрати пам'яті (найчастіше використання пам'яті заборонено), швидкодії (1-2 такту) і площі (кілька сотень FPGA- або ASIC- комірок). Через такі суворі вимоги до апаратних ГПСЧ є проблема у створенні генератора,який буде криптостійким, тому до даного моменту всі відомі апаратні ГПСП були зламані.

ГПСП на основі криптографічних алгоритмів

Безпечний блоковий шифр можна перетворити в ГПСП, запустивши його в режимі лічильника. Таким чином, вибравши випадковий ключ, можна отримувати наступний випадковий блок застосовуючи алгоритм до послідовних натуральних чисел. Рахунок можна починати з довільного натурального числа. Очевидно, що безпека такої схеми повністю залежить від таємності ключа.

Звернемо увагу на алгоритми потокового шифрування.

Алгоритми потокового шифрування засновані на гамуванні закритого потоку даних псевдовипадковою послідовністю (ПСП). Гамуванням називається спосіб шифрування, при якому на відкритий текст накладається гамма-послідовність, наприклад, за допомогою операції побітового «виключає або» (XOR), а повторне застосування операції до тексту дає вихідне відкрите повідомлення.

Таким чином, акцентувати увагу слід на алгоритмі генерації ПСП, оскільки саме від його властивостей буде залежати практична застосовність методу захисту потокових даних.

Одним з класичних алгоритмів генерації ПСП, використовуваних для потокового шифрування, є алгоритм RC4, який був розроблений в 1987 році Рональдом Лінном Ривестом, відомим американським фахівцем в області криптографії, який також є співавтором алгоритму RSA, автором хеш-функції MD5 і цілого ряду інших активно використовуваних в криптографії хеш-функцій і шифрів. Протягом семи років шифр був комерційною таємницею , і точний опис алгоритму надавалося тільки після підписання угоди про нерозголошення, але у вересні 1994 року його опис було анонімно відправлено в розсилку Cypherpunks. Незабаром опис RC4 було опубліковано в новинній групі sci.crypt. Ця група і поширила вихідний код на всю мережу. Цей шифр показував ті ж результати на виході, які давав справжній RC4. Опублікований шифр сумісний з наявними продуктами, які використовують RC4; власники легальних копій RC4 заявили про збіжність алгоритмів при різниці в позначенні та структурі програми.

Алгоритм вже не являвся секретним. Але сама назва «RC4» так і залишилася маркою компанії RSA.

Деякі шифрування і стандарти все ж використовують даний алгоритм(WEP,WPAiTLS).

Головні чесноти RC4- простота його реалізації і висока швидкість роботи як у програмній частині так і в апаратній.

Довжина ключа в США рекомендована для використання всередині країни дорівнює 128 бітам, але у RC4 за рахунок угоди є можливість експортувати шифри,у яких довжина ключа буде до 40 біт.

Алгоритм має можливість модифікації на випадок роботи не з байтами, а з елементами довільної розрядності. Саме така модифікація і буде розглянута в подальшому.

Наведемо опис класичної версії алгоритму.

Алгоритм складається з двох стадій: генерації початкового стану і власне породження псевдовипадкової послідовності.

Як уже було згадано, алгоритм RC4 в даний час надзвичайно широко застосовується в криптографічних завданнях. Можна стверджувати, що він залишиться актуальним і в найближчому майбутньому, цьому є ряд передумов: по-перше, широке поширення пристроїв, що використовують шифрування на його основі, заміна яких в даний час не передбачається; по-друге - характерне для нього поєднання порівняно високої обчислювальної ефективності з хорошими статистичними показниками, наприклад, генерується надзвичайно великий за довжиною періоду ПСП.

Ця сукупність обставин зробила його останнім часом об'єктом пильної уваги з боку провідних фахівців з криптографії, опублікований ряд робіт, що стосуються як різного роду недоліків алгоритму, так і способів їх подолання. Проте, пропонований спосіб дослідження алгоритму через його модифіковану форму ще ніде не розглядався.

1.2 Квантова криптографія

Одним з напрямків квантової інформації, найбільш просунутої в область практичних застосувань, є квантова криптографія. Завдання криптографії полягає в тому, щоб передати певну секретну інформацію від джерела (Аліса) до приймача (Боб) так, щоб спроба шпигуном (Єва) перехопити передачу або дізнатися секретний код була приречена на провал.

Сучасними методами класичної криптографії це завдання майже вирішується, наприклад, в рамках симетричної криптосистеми, що спирається на створення секретного коду. У цій системі Аліса і Боб мають секретний код (К) - послідовність випадкових чисел (наприклад: десяткових), довжина якої дорівнює довжині кодованого повідомлення. Безпека цієї криптосистеми полягає в повній секретності цієї послідовності. По заданому правилу, кожній букві алфавіту ставиться у відповідність десяткове число, і Аліса, замінює в посланні (П) кожну букву відповідної їй цифрою.

Само по собі, таке зашифроване послання легко дешифрується. Потім послідовність шифрується, тобто до кожного числа послання додається число з коду (К) і виходить, що цифри в розряді одиниць представляють собою криптограму ($К_р$), яку можна пересилати з публічного каналу (телефон, кур'єр, Internet і т.д.). Боб, отримавши криптограму ($К_р$), розшифровує її, використовуючи секретний код (К), і отримує, таким чином, передане йому повідомлення (П).

Такий спосіб кодування був винайдений в 1918 році Г.Вернамом (Gilbert Vernam) з американської телеграфної компанії і майором американської армії Дж.Мауборне (Joseph Mauborgne). Цей перший абсолютно секретний спосіб передачі інформації в подальшому було названо шифром Вернама.

У 1949 році С.Шеннон, спираючись на розроблену ним теорію інформації, довів теорему, що дана криптосистема є абсолютно секретна, якщо секретний код - істинно випадковий, довжина коду не менша за довжину кодованого повідомлення, і цей код використовується для передачі інформації тільки один раз. [6]

Однак на практиці реалізація даної системи наштовхується на серйозні труднощі. Одна з них - це створення і передача великого секретного коду, необхідного кожен раз, коли потрібна передача нової інформації. Уникнути цієї складності, можна було б при наявності швидкісного фізичного каналу, секретність якого забезпечувалася б фізичними законами. Саме такий канал і являє квантова фізика. Головне правило квантової криптографії-окремий об'єкт(квантовий) не клонується.

11

Клонування в даному випадку означає генерування копії вихідного об'єкта при цьому зберігається той стан,який був до початку генерування і яке спочатку невідомо.

11

Отже, якщо в якості передавача секретного коду виступають стани окремих частинок, то при спробі зареєструвати ці стани зовнішнім спостерігачем, вони руйнуються.

В даний час розроблено безліч протоколів, які дозволяють виявити спробу перехоплення квантового каналу. Один з таких протоколів пов'язаний з кодуванням поляризаційних станів фотонів в двох альтернативних базисах, що не ортогональних один одному.

У зарубіжній літературі цей протокол позначається як BB84. [4,5,6,7] Кожна з цифр кодується двома поляризаціями для досягнення гарантії таємності. Якщо використовувати тільки один базис, то залишається тільки квантовий канал для передачі коду від Аліси до Боба. Але в цьому випадку, навіть якщо Аліса передасть випадковий код, у Боба не буде можливості перевірити істинний він або зіпсований спробами перехоплення.

Пояснимо на конкретному прикладі процес обміну секретним ключем між Алісою і Бобом.

1. Перш за все, Аліса і Боб домовляються про кодування: фотони з поляризацією 0 і 45 закодовують «0», а фотони у яких поляризація 90 і 135- одиницю. Причому ця угода може не бути секретним, і, припустимо, що Єві вони відомі. Потім Аліса випадково змінює поляризацію фотонів,які відправляються квантовим каналом до Бобу.

Зазвичай для досягнення більшої секретності в якості генератора випадкових чисел можна використати шум діода, але ні в якому випадку не комп'ютерний генератор, тому що Єва може змоделювати такий генератор і отримати схожу послідовність з дуже маленькою помилкою або взагалі без неї. Аліса повинна записувати послідовність, щоб надалі можна було відредагувати код, отриманий Бобом, і виявити наявність Єви. Ця послідовність є секретною для всіх.

12

Нехай, наприклад, Аліса передала послідовність. З протилежного боку квантового каналу Боб вимірює поляризацію одержуваних фотонів, 12 випадковим чином змінюючи орієнтацію аналізатора для розпізнавання поляризації 0 і 90 (+) або 45 і 135 (-). Як було сказано вище, відповідно до теорії інформації С.Шеннона, розглянута криптосистема буде секретною тільки в тому випадку, якщо використовується код випадковий. Саме

випадковості коду і домагаються Аліса і Боб, випадково змінюючи поляризатори. Боб також фіксує отриману у нього послідовність, і вона теж повинна бути секретною.

Припустимо, Боб підставляв аналізатори наступним чином і отримав певну послідовність поляризаційних станів фотонів.

Наприклад, Аліса передала послідовність.

З протилежного боку квантового каналу Боб вимірює поляризацію одержуваних фотонів, випадковим чином змінюючи орієнтацію аналізатора для розпізнавання поляризації.

Використовуючи публічний канал, Боб повідомляє Алісі, який тип вимірювання він виконував для кожного фотона (не результат вимірювання), а Аліса підтверджує, правильно чи ні він вибрав його.

Висновки

У огляді науково технічної літератури були приведені різноманітні види генераторів псевдовипадкових послідовностей. Після визначення переваг та недоліків даних видів визначено,що саме генератори псевдовипадкових послідовностей на основі криптографічних алгоритмів є одним із найкращих рішень для забезпечення підвищення захисту інформації.

Також у огляді описано принцип роботи квантової криптографії на простому прикладі обміну секретним ключем між двома користувачами.

2. Теоретична частина

2.1 Методи захисту інформації

Сучасна криптографія потрібна для (де)шифрування інформації ,також вона виконує такі функції, як: автентифікація, контроль цілісності інформації, незаперечність причетності до авторства чи до одержання документа або повідомлення. Визначимо автентифікацію як процедуру перевірки належності учаснику процесу обміну інформацією ідентифікатора, який був їм пред'явлений. Ідентифікатор – це послідовність латинських літер і цифр, яка починається з літери. Цілісність інформації – це властивість інформації, яке полягає в тому, що вона не може бути модифікована неавторизованим користувачем і/чи процесом.

Незаперечність причетності до авторства – це поняття, зворотне поняттю відмови від авторства, тобто заперечення причетності до утворення або передавання якого-небудь документа чи повідомлення. В свою чергу, незаперечність причетності до одержання документа, або повідомлення – поняття, зворотне поняттю відмови від причетності до утворення або передавання якого-небудь документа чи повідомлення. Важливим поняттям криптографії є криптоаналітична атака– це загальна назва методу, яким криптоаналітик намагається зламати криптосистему. У залежності від якості та кількості інформації, якою володіє криптоаналітик, криптоаналітичні атаки поділяють на атаки лише із криптотекстом, атака з відомим відкритим текстом, атака з вибраним відкритим текстом, атака з вибраним криптотекстом, атака з вибраним ключем.

Важливим елементом криптографії є криптостійкість алгоритму,тобто це стійкість алгоритму до розшифковки,іншими словами до взлому.Існує абсолютна та практична крипостійкість. Різниця між ними у тому,що практична стійкість може бути дешифрованою через якийсь час у абсолютній стійкості довжина ключа є не меншою від довжини відкритого тексту та ключ обирається з ключового простору дійсно випадково.Криптографічна система складається із криптографічних методів. Криптографічна система, або криптосистема– це сукупність програмних, апаратних, програмно-апаратних засобів, криптографічних алгоритмів або криптографічних схем,

поєднаних в єдиній системі з метою розв’язання конкретної задачі захисту інформації.

Початкова криптографія була побудована на використанні криптоалгоритмів, які користувачі мали тримати в таємниці. Розкриття цих алгоритмів третій стороні автоматично приводило до розкриття шифротексту. Такі криптоалгоритми з часом отримали назву тайнопису. Тайнопис– це методи кодування, особливістю яких є обов’язкове збереження в таємниці криптографічного алгоритму від третьої сторони. Але їх можливості є досить обмежені тому наразі вони не використовуються.

Наприкінці XIX сторіччя криптографія починає відмовлятися від тайнопису і переходити до криптоалгоритмів з ключем. Такі зміни пов’язують з ідеями французького математика Огюста Кергоффса, сформульованими в роботі “Військова криптографія”, яка вийшла в світ у 1883 році. В цій роботі О.Кергоффс вперше відокремлює проблеми таємності криптоалгоритму і таємності ключа. Він пропонує використовувати криптоалгоритми, для яких пріоритетним є таємність ключа і не є важливою, власне, таємність криптоалгоритму.

Саме на цьому і будується сучасна криптографія-на шифруваннях з ключем. Шифрування з ключем– це методи кодування, особливістю яких є обов’язкове збереження ключа в таємниці від третьої сторони, збереження криптографічного алгоритму при цьому не обов’язково.

Серед криптоалгоритмів шифрування з ключем бувають симетричні та асиметричні.Різниця між ними полягає в тому, що у симетричному шифруванні використовується один спільний ключ ,в асиметричному-використовується 2 ключі приватний і відкритий,цей метод ще називається– шифрування з відкритим ключем

Симетричне та асиметричне шифрування є домінуючими криптографічними методами на поточний час. Історія криптографії із симетричними криптоалгоритмами почалась наприкінці XIX сторіччя з виходом в світ роботи О. Кергоффса “Військова криптографія”.

Криптографія з асиметричними криптоалгоритмами значно молодше. Її початок пов'язують з роботами американських вчених В.Діффі і М.Хеллмана, які вийшли в світ у 70-х роках минулого сторіччя.

Симетричні та асиметричні криптоалгоритми мають свої переваги та недоліки. Симетричні криптоалгоритми в порівнянні з асиметричними мають більшу швидкодію та меншу довжину ключа. Асиметричні криптоалгоритми, в свою чергу, можуть застосовуватися у таких випадках організації криптосистем, де використання симетричних алгоритмів є неможливим. Порівняння характеристик цих криптоалгоритмів, в цілому, не є коректним, тому що вони створені для розв'язання різних завдань шифрування.

Асиметричні криптоалгоритми, як і симетричні, застосовуються для шифрування масивів даних, але їх швидкість значно поступається швидкості останніх. Основне призначення асиметричних криптоалгоритмів – забезпечення ефективного функціонування сучасних криптосистем. Ці криптоалгоритми складають основу задач автентифікації користувачів, контролю цілісності інформації, незаперечності причетності до авторства чи до одержання документа або повідомлення та інших.

Практично асиметричні крипто алгоритми використовуються у електронно-цифровому підпису (ЕЦП) криптографічних систем. Електронно-цифровий підпис– цифрова послідовність, що додається до повідомлення (даних) для забезпечення цілісності та підтвердження авторства і формується із застосуванням асиметричних криптосистем. У електронно-цифровому підписі для шифрування повідомлень використовується закритий, а для розшифрування – відкритий ключ.

2.2 Критерії якості псевдовипадкових послідовностей

Існує дуже багато класів алгоритмів генерації ПСП, що застосовуються на практиці. Слід, однак, зауважити, що способи

використання ПСП досить різноманітні, і вони пред'являють різні вимоги до якостей послідовності. Цікавим в рамках цієї роботи є використання в криптографії як гамма-послідовності, з якою точки зору ми і будемо підходити до аналізу пропонованої модифікації RC4.

Першим кроком аналізу криптоалгоритму або криптосистеми є побудова моделі атакуючої системи; в криптографії загальноприйнятий ряд припущень щодо цієї моделі. Зокрема, прийнято, що атакуючий має можливість подавати на вхід досліджуваного алгоритму свій відкритий текст і аналізувати одержувані шифрограми. Стосовно до алгоритму гамування це означає, що атакуючий має можливість, подаючи на вхід повідомлення, що складаються з нульових байтів, отримати чисту гамма-послідовність і досліджувати її властивості. Ця обставина дозволяє висунути ряд вимог до гамма-послідовності, наприклад:

- ПСП повинна володіти великим періодом. Справді, якщо атакуючому вдається виявити період гамма-послідовності, він здатний генерувати її самостійно на підставі отриманих даних, тим самим отримуючи доступ до зашифрованих даних.
- Атакуючий не повинен бути в змозі передбачити статистичні властивості послідовності, зокрема, кількість одиничних і нульових бітів не повинно бути сильно різним; розподіл частот бітових слів різної довжини має бути рівномірним. Детальніше це питання розглянуто в розділі «Тестування ПСП»

Відома оцінка зверху періоду ПСП, яку породжена класичним варіантом RC4; вона перевищує 10^{500}. Величина такого порядку повністю виключає можливість виявлення періоду в послідовності атакуючим. Однак, актуальним є питання про досяжність цієї верхньої оцінки і, взагалі, про характерні довжини періодів. До цього питання можна підійти як з теоретичного боку, дослідивши алгоритм породження, так і від практики, вивчивши поведінку самої послідовності. З огляду на величезні значення

періоду класичної варіації, другий підхід до неї непридатний. Його застосування стало можливим після введення пропонованої модифікації.

Розглянемо способи визначення якостей ПСП, які породжені аналізованим алгоритмом. Серед використовуваних для цього тестів можна виділити дві великі категорії.

Перша з них - графічні тести. До цієї категорії відносяться тести, результати яких відображаються у вигляді графіків, що характеризують властивості досліджуваної послідовності. Серед них:

- гістограма розподілу елементів послідовності: дозволяє оцінити рівномірність розподілу символів в послідовності і визначити частоту повторення кожного символу;

- розподіл на площині: призначене для визначення залежності між елементами послідовності;

- перевірка серій: дозволяє визначити рівномірність окремих символів в послідовності, а так само рівномірність розподілу серій з k біт;

- перевірка на монотонність: служить для визначення рівномірності виходячи з аналізу незростаюча і неубутних підпослідовностей;

- автокореляційна функція: призначена для оцінки кореляції між зсунутими копіями послідовностей і окремих підпослідовностей;

- профіль лінійної складності: тест оцінює залежність лінійної складності послідовності від її довжини;

- графічний спектральний тест: дозволяє оцінити рівномірність розподілу біт послідовності на підставі аналізу висоти викидів перетворення Фур'є.

Результати графічних тестів інтерпретуються людиною, тому на їх основі висновки можуть бути неоднозначними.

Незважаючи на загальноприйняту назву, деякі графічні тести можуть застосовуватися для отримання чисельної характеристики якості послідовності. Наприклад, за допомогою тесту на розподіл елементів послідовності можна отримати середню ентропію, з теста на монотонність -

середню довжину монотонної послідовності. Проблема суб'єктивності інтерпретації результатів такого тесту, проте, залишається.

Друга категорія тестів ПСП - статистичні тести.

Тести цієї категорії дозволяють отримати об'єктивну чисельну характеристику послідовності, на підставі якої можна зробити висновок про придатність генератора ПСП до застосування. Для цього, як правило, формулюється деяке твердження про властивості ідеальної випадкової послідовності, що мають чисельне вираження. Потім обчислюється відповідне значення для досліджуваної послідовності. Якщо воно занадто сильно відрізняється від очікуваного, то гіпотеза про випадковость досліджуваної послідовності відкидається. Розглянемо цю процедуру на простому прикладі.

Клод Шеннон довів, що в разі, коли гамма-послідовність має рівний розподіл нулів і одиниць (тобто ймовірність появи нульового або одиничного біта дорівнює 0,5), шифротекст також має рівні ймовірності появи нуля і одиниці, незалежно від їх розподілу в вихідному відкритому тексті. Таким чином, досягається приховування статистичних властивостей вихідного тексту, що є бажаним. Таким чином, можна сформулювати вимогу до випадкової послідовності, що використовується в якості гами, щоб кількість нулів і одиниць в ній було приблизно однаковим.

Обчисливши реальне співвідношення кількостей нулів і одиниць у досліджуваній послідовності, можна стверджувати, що вона є локально випадкової, якщо це співвідношення відрізняється від очікуваного 0,5 на відповідну довжину дослідженої ділянки малу величину, що існує, як квадрат цієї довжини, і не є - в іншому випадку.

Слід, однак, зауважити, що локальна випадковість не дає гарантії високої якості генератора ПСП як такого. Не виключений випадок, коли прийняте допущення справедливо на конкретній ділянці послідовності і не виконується на інших ділянках. Ця проблема нівелюється повторними запусками тесту і збільшенням довжини досліджуваної послідовності. Інша

проблема пов'язана з тим, що при деяких умовах якісна послідовність може не пройти тест. Проілюструвати це можна виродженим прикладом, коли на вхід описаного вище тесту подається послідовність з одного біта: розподіл гарантовано буде «поганим». У випадку з таким тривіальним тестом цього легко уникнути, але коли розглядаються більш складні, слід мати на увазі, що невелика кількість провалених тестів не дає підстави стверджувати непридатність генератора,

Нарешті, той чи інший алгоритм генерації ПСП може задовольняти критеріям окремо взятого тесту, але бути непридатним до використання в криптографії. Згаданому вище критерієм однакової кількості нулів і одиниць в потоці біт задовольняють практично всі генератори, в тому числі, наприклад, лінійний конгруетний, який має істотну кількість інших недоліків, які роблять його непридатним для використання в якості генератора гами. Ці недоліки можна виявити іншими тестами.

До будь-якого способу перевірки ПСП можна придумати послідовність, що задовольняє його вимогам, але тим не менше володіє істотними недоліками. Тому тільки комплексне тестування на відповідність багатьом різним критеріям може вважатися достатньою підставою для твердження про близькість досліджуваної послідовності до випадкової.

Існує ряд пакетів для такого комплексного тестування, кожен з яких включає до декількох десятків різнорідних тестів. Серед них можна назвати добірку тестів Д. Кнута, пакет STS Національного інституту стандартів і технологій (NIST, США), DIEHARD.

2.3 Модифікації алгоритмів

Модифікація з довжиною блоку n = 3.

Для даної модифікації існує 40320 можливих початкових перестановок S. Теоретична довжина періоду кожної ПСП не перевищує 2580480 елементів.

Для експерименту використовувалася вибірка з 1000 ПСП, одержуваних з 1000 різних початкових станів блоку перестановок.

Результати виявилися наступними:

1. 757 послідовностей мають період 955496 елементів;
2. 141 послідовність має період 322 120 елементів;
3. 44 послідовності мають період 29032 елемента;
4. 19 послідовностей мають період 53000 елементів;
5. 16 послідовностей мають період 44264 елемента;
6. 15 послідовностей мають період 4696 елементів;
7. 4 послідовності мають період 9624 елемента;
8. 1 послідовність має період 472 елемента;
9. 1 послідовність має період 3008 елементів;
10. 1 послідовність має період 9432 елемента;
11. 1 послідовність має період 456 елементів;

Як видно з результатів, жодна з 1000 ПСП не досягає максимально можливого періоду, при цьому мінімальний період в даній вибірці дорівнює 456 елементів [10].

Для тестування пакетом NIST використовувалася вибірка з 100 послідовностей. Модифікація пройшла 2 тести з 15 повністю та кілька підтестів тестів Non-Overlapping Template Matching Test і Random Excursions Variant Test. В цілому результати виявилися кращими модифікації з n = 2, але недостатньо задовільні для практичного застосування [10].

Модифікація з довжиною блоку n = 3.

Для модифікацій алгоритму з довжиною блоку n = 3 і вище також були перевірені вибірки з 1000 послідовностей, кожна розміром 1000 000 байт. Ні в однієї з них не був знайдений період, що дозволяє стверджувати, що періоди протестованих ПСП більше або дорівнюють 1000 000 байт.

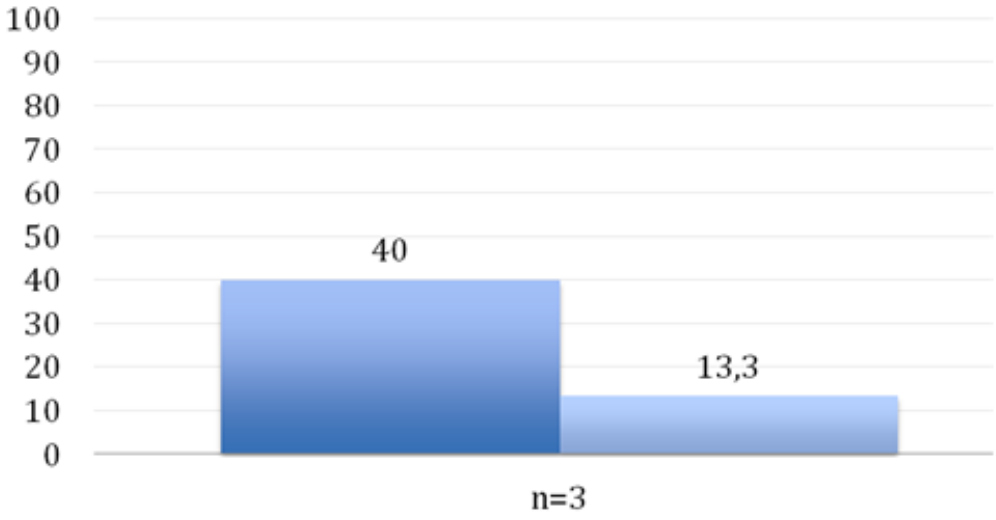


Рисунок 2.1 – Результати модифікації алгоритму

Однак, це не дає підстави припускати, що для модифікацій алгоритму з довжиною блоку n = 3 не існує послідовностей з аномально коротким періодом.

Модифікація з n = 4 показала суттєво кращі результати при тестуванні пакетом NIST, ніж попередня. Вибірка з 100 ПСП пройшла всі тести, за винятком 2 підтестів тесту Non-Overlapping Template Matching Test, 2 підтестів тесту Random Excursions Test і 1 підтесту Random Excursions Variant Test.

Вибірка з 100 ПСП модифікації з n = 5 пройшла всі тести пакету NIST.

Вибірка з 100 ПСП модифікації з n = 6 пройшла всі тести пакету NIST, за винятком 1 підтесту Non-Overlapping Template Matching Test і 1 підтесту Random Excursions Variant Test.

Вибірка з 100 ПСП модифікації з n = 7 пройшла всі тести пакету NIST, за винятком 1 підтесту Non-Overlapping Template Matching Test.

Було також проведено тестування модифікацій алгоритму з n = 8 (класична версія) і n = 16. Обидві модифікації успішно пройшли тестування, причому версія з n = 16 не показала суттєво кращих результатів у порівнянні з n = 8 (рисунок 2.2).

Діаграма на рисунку 2.2 демонструє відсоток успішно пройдених тестів разом з підтестами і відсоток успішно пройдених тестів для п'яти модифікацій алгоритму (з n від 4 до 8).

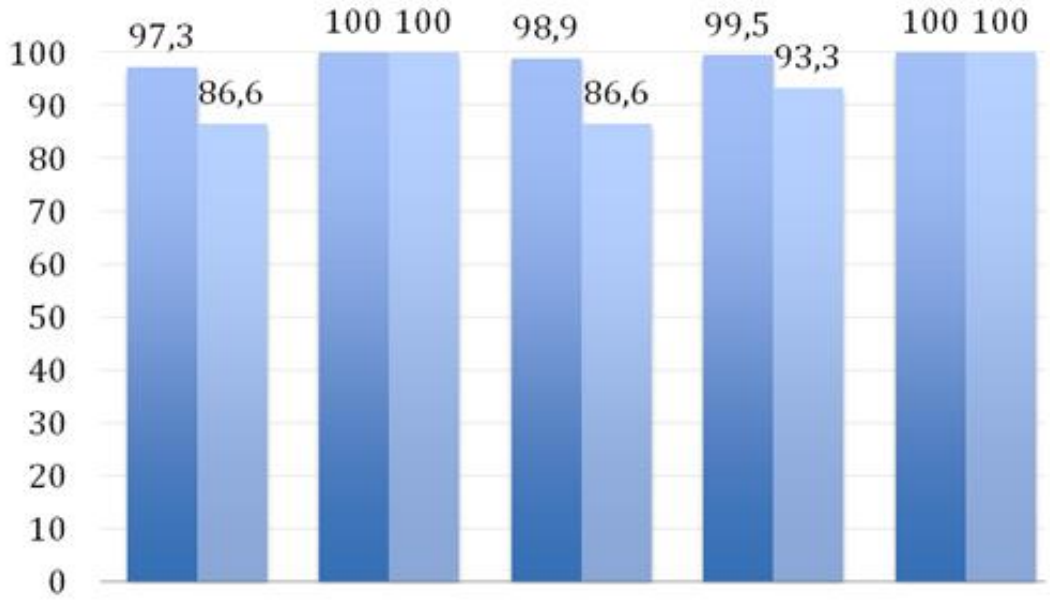


Рисунок 2.2 – Результати модифікації

Висновки

В результаті вивчення матеріалів аналітичного огляду,виділена квантова криптографія, як один із найкращих способів по захисті інформації.

Псевдовипадкові послідовності на основі гамма-послідовністіє найкращим варіантом, який можна використати для створення алгоритму та виконати модифікацію елементів довільної розрядності . На підставі наведеної в аналітичному огляді інформації, ретельного розгляду ключових і найбільш значущих моментів даної теми та дослідження результатів модифікацій алгоритму складемо структурну схему пристрою

3. Розробка Стартап проекту

Стартап-проект – це тип бізнесу, головна ціль-це отримання прибутку від реалізації нової ідеї. Із появою Інтернету як інструменту комунікацій та збуту товару, набагато легче шукати нових інвесторів та покупців вашого товару. Але при реалізації стартап-проекту можна зіштовхнутися з рядом проблем ,які можуть привести до повного краху вашого проекту/ідеї.. Причини можуть бути наступні:

1. Погана презентація проекту/ідеї перед інвесторами(вкладниками).
2. Не потрібність продукту на сучасному ринку.
3. Невірний розподіл грошей та силового ресурсу, тобто банкротство проекту та інші.

Оцінюючи теперішній ринок, шанс впровадження власного стартапу приблизно 15%. Головна перевага стартапу – його ідея. Проекти,які можуть досягти успіху(отримати дохід) обладають такими характеристиками:

1. Проста ідея для реалізації.
2. На створення продукції не потрібно витрачати багато часу.
3. Низька собівартість.
4. Продукція буде новою,не мати аналогів.

Етапи розробки стартап-проекту:

1. **Маркетинговий аналіз стартап-проекту.**

В цьому пункті досліджується ідея проекту та де будуть головні напрями реалізації товару. Головне на цьому етапі – це визначення відмінностей свого проекту від проекта своїх конкурентів. Наступним крок- провести аналіз можливого успішного впровадження свого продукту на ринок збуту. На всі ці

дії створюються стратегія,якої необхідно буде дотимуватися для успішного поширення продукту на ринку.

2. Організація стартап-проекту.

Цей пункт відповідає за створення На цьому етапі розробляється план-графік, згідно якому буде впроваджена найшвидша та найуспішніша програма для стартап-проекту.

Знаходяться канали,по яким буде відбуватися придбання апаратури ,щоб створювати продукт. Наступний крок –обчислення скільки потрібно матеріального та людського ресурсу для впровадження першої партії продукту. І заключний крок даного пункту – обчислення витрат для старту проекту.

3. Фінансово-економічний аналіз та оцінка ризиків проекту.

Для вдалої презентації та старту стартап-проекту потрібен основний обсяг інвестиційних витрат компанії. Для того, щоб інвестори були зацікавлені в даному проекті потрібно визначити і звернути свою увагу наскільки цей проект буде привабливим для інвесторів.Тобото,розглянути,відповідає стартап таким умовам, як:

- Час,за який проект окупиться.
- Який є запас фінансової міцності.
- Наскільки це рентабельно.

Також інвестори звертають увагу на ризики проекту і дуже важливо знайти одразу можливі шляхи запобігання і,на всяк випадок,вирушення даних ризиків і представити їх інвесторам.

4. Заходи з комерціалізації проекту.

Після підготовки до презентації стартап-проекту,відбувається знаходження інвесторів та просування оферти. Під цим розуміється: визначення фокусу компанії на певний відділ інвесторів та виявлення

зацікавленості інвесторів в власному проекті компанії. Наступний етап - створення оферти:

- Коротка характеристика проекту,щоб інвестори мали смогу швидко дізнатися про власний стартап .

Теж важливо у цьому пункті плунавтита відвідувати різні заходи по презентації вашого проекту,де з ним зможуть ознайомитися не тільки інвестори, а й можливо перші покупці:

- Впровадження комунікаційних каналів та створення заходів ,де буде відбуватися презентація проекту;

Виконання всіх пунктів,їх правильна та вчасна реалізація дають можливість для вдалого старту стартапу.

3.1. Опис ідеї проекту

Опис ідеї стартап-проекту наведено нижче в табл. 3.1. В цій таблиці наведені сильні та слабкі сторони проекту.

Табл. 3.1 Опис ідеї стартап-проекту

Зміст ідеї	Напрямки застосування	Вигоди для користувача
Зміст ідеї: Створення кращого методу захисту інформації для ПК.	1. Технічна галузь	Розробка власних пристроїв та контроль виробництва пристроїв на виробництві. Висока криптостійкість

		системи. Доступність та розвиток приладу.
	2. Промислова галузь	Контроль стійкості виробничих приладів, та вдосконалення у подальшому як програмного коду, так і апаратної частини.

Табл. 3.2 Слабкі та сильні сторони власного стартап-проекту та конкурентів

Технічно-економічні характеристики ідеї	Товари та концепції конкурентів				W Слабка сторона	N Нейтральна сторона	S Сильна сторона
	Власний проект	Конк. № 1	Конк. № 2	Конк. № 3			
Захист від можливого взлому простим користувачем	Так	Так	Так	Так			+
Захист від взлому шляхом автоматичного підбору ключів	Так	Ні	Ні	Ні			+
Захист від взлому у випадку внутрішнього проникнення в систему	Ні	Ні	Ні	Ні	+		

На основі таблиці 3.2 про слабкі та сильні сторони власний стартап-проект має переваги на ринку серед конкурентів, та базуючись на табл. 3.1 можна виявити основні шляхи для впровадження даного стартап-проекту.

3.2. Технічний аудит ідеї проекту.

Важлива частина технічного проекту в компанії на базі якого планується розробити стартап-проект являється технічна здійсненність ідеї проекту що наведено в табл. 3.2.1

Табл. 3.2.1 Технологічна здійсненність ідеї стартап-проекту.

Ідея проекту	Технології її реалізації	Наявність технології	Доступність технології
Розробка приладу для забезпечення захисту інформації,створення спеціально модифікованого програмного коду та втілення інноваційних ідей для покращення апаратного забезпечення.	Розробка, вдосконалення шляхом постійних оновлень, програмування ;	Наявна;	Доступна;

За даними з таблиці 3.2.1 можна зробити висновок, що даний прилад має технологічну реалізації на сучасному ринку, існують наявні та доступні технології для виробництва та реалізації ідеї даного стартапу.

3.3. Аналіз ринкових можливостей запуску стартап-проекту

Головне на що звертають увагу інвестори-це картина ринку, та яким чином продукт компанії,в яку вони інвестують,буде конкурентним і як буде відбуватися продаж товару у вже сформованому ринку.

Табл. 3.3.1 Характеристика постійних клієнтів

Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
Покращений метод захисту інформації.	1) Великі та малі компанії,які виступають за збереження секретності як власних даних,так і персональних даних своїх співробітників. 2) Звичайні користувачі мережі Інтернет.	Об'єм купівлі та складність використовування приладу.	До продукту: • Якість; • Надійність; • Швидкодія; До компанії: • Сервісна підтримка; • Виробництво нових модулів для захисті від нових видів вірусних атак; • Розробка ПО для нових модулів;

Наступним кроком буде розпис факторів загроз та їх детальній зміст, а також розпис реакції та дії компанії для захисту від цих загроз. Ця вся інформація наведена в табл. 3.3.2

Табл. 3.3.2 Фактори загроз та реакція компанії на них

Фактор	Зміст загрози	Реакція компанії
1) Конкуренція.	Одна із головних загроз на ринку – конкуренція. Одною із проблем являється поставлення своєї продукції кращою	Розробка стратегії на цільовому ринку таким чином, щоб представити плюси власного виробу споживачу та розробити

	ніж інший виробник на тому самому ринку.	стратегію для утримання клієнтів.

Наведемо фактори можливостей власного стартап-проекту на ринку, приведені результаті вписані в табл. 3.3.3.

Табл. 3.3.3. Фактори можливостей стартап-проекту

Фактор	**Зміст можливостей**	**Можлива реакція компанії**
1) Можливість додавання нових модулів для вдосконалення апаратної частини приладу.	Змога до додавання додаткових модулів за рахунок чого буде підвищуватися не тільки криптостійкість системи але і її швидкодія.	Залучення нових клієнтів через поширення даної можливості на можливих каналах маркетингу.
2) Можливість модифікації програмного коду.	Можливість до наявного коду вносити поправки та зміни,для покращення рівня захисту від нових видів кібератак.	Дану можливість можливо пропанувати компаніям, які працюють з великою кількості інформації,яка потребує конфіденційності.

Характеристику ринку та аналіз конкуренції можна представити в табл. 3.3.4.

Табл. 3.3.4. Ступеневий аналіз конкуренції на ринку.

Особливості конкурентного середовища	**В чому проявляється дана характеристика**	**Вплив на діяльність підприємства**
1. Тип Конкуренції: чиста	В кого кращий товар, чи послуга – в того і купують.	Покращення якості товару на ринку.

2. За рівнем конкурентної боротьби: Світова	Продукт проекту належить до звуженого ринку збуту.	Додавання функціоналу та орієнтації споживачів.
3. За галузевою ознакою: одногалузева	Може бути використана в одній із галузей.	Покращення якості в одній із галузей.
4. Конкуренція за видами товарів: Товарно-видова	Відрізняється функцією додавання додаткових модулів для покращення функціональності системи.	Додавання відрізняючих від конкурентів функцій.
5. За характером конкурентних переваг: цінова та нецінова	Продукт привабливіший – чим дешевше. Чим продукт якісніший – Тим рентабельніше.	Покращення цін та якості товару у порівнянні з конкурентами.

Головним фактором для споживача являється – ціна та якість. В наступній табл. 3.3.5 записані фактори конкурентоспроможності на основі головних факторів для споживача.

Табл. 3.3.5. Фактори конкурентоспроможності.

Фактор конкурентоспроможності	**Обґрунтування**
1) Ціна	На ринку зі схожими приладами споживач обиратиме ту, що являється найдешевшою.
2) Якість	На ринку зі схожою ціновою політикою між конкурентами споживач обиратиме той продукт – який має найкращі характеристики.
3) Відомість	При рівності таких параметрів як ціна та якість споживач обертатиме товар який найбільш відомий на ринку.

За даними факторами споживач буде обирати для себе найбільш привабливий продукт на ринку в залежності від пріоритетів. Тож важливо вигравати по всіх факторах конкурентноспроможності у конкурентів на цільовому ринку.

Маючи всю важливу інформацію про цільовий ринок, та споживачів в цьому ринку, можна провести порівняльній аналіз сильних та слабких сторін, які представлені в табл. 3.3.6.

Табл. 3.3.6. Порівняльний аналіз сильних та слабких сторін

Фактор Конкурентоспроможності	**Бали 1 – 20**	**Рейтинг товарів-конкурентів у порівнянні з "DP"**						
		-3	**-2**	**-1**	**0**	**+1**	**+2**	**+3**
Ціна					+			
Якість					+			
Відомість				+				

Підсумувати інформаційний аналіз можна SWOT-аналізом стартап-проекту який приведений в табл. 3.3.7.

Табл. 3.3.7. SWOT-аналіз стартап-проекту

Сильні сторони: Висока криптостійкість; Швидкодія;	Слабкі сторони: Шанс злому новим видом вірусу
Можливості: Можливість додавати або змінювати модулі в апаратній частині; Можливість вдосконалення програмного коду;	Загрози: Конкуренти;

В таблиці 3.3.7. представлені сильні та слабкі сторони проекту, а також можливості та загрози.

3.4. Розробка маркетингової програми стартап-проекту

Щоб забезпечити успішний старт стартапу необхідна маркетингова програма. За допомогою даної програми можна створити стратегію конкурентної поведінки. Результати приведені в табл. 3.4.1.

Табл. 3.4.1 Визначення базової стратегії конкурентної поведінки на цільовому ринку.

Чи являється проект першопроходцем на цільовому ринку?	**Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?**	**Чи буде компанія копіювати основні характеристики товару конкурента, і які?**	**Стратегія конкурентної поведінки**
Якщо розглядати прилад з структурної точки зору – ні. Аналоги даного приладу існували в минулому, але в приладі в даному стартап-проекта можна побачити покращення в криптостійкості та швидкодії, а також в нових можливостях, яких не має в конкурентів на цільовому ринку.	Стратегія компанії побудована на принципі, що клієнти будуть переходити від конкурентів за рахунок нижчої ціни, а також за рахунок вищої якості.	Так – Загальні принципи для покращення характеристик приладу.	Агресивна стратегія конкурентної поведінки.

Визначимо ключові переваги концепції для потенційного товару. Визначимо ці параметри в наступній табл. 3.4.2

Табл. 3.4.2. Визначення ключових переваг концепції потенційного товару

Потреба	**Вигода, яку пропонує товар**	**Ключові переваги перед конкурентами**
1) Підвищення захисту інформації у ПК.	Високу криптостійкість за рахунок використання псевдовипадкових	Використання потужнішого генератора ніж у конкурентів.

	послідовностей.	
2) Висока швидкодія	Високу швидкодію обробки інформації за рахунок використання мікросхем.	Використання мікросхем,що спрощують принципову схему та схему включення.

Важливою частиною в продажу товару на цільовому ринку – система збуту. Опишемо цю саму систему в табл. 3.4.3.

Табл. 3.4.3. Формування системи збуту

Специфікація закупівельної поведінки цільових клієнтів	**Функції збуту, які саме має виконувати постачальник товару**	**Глибина каналу збуту**	**Оптимальна система збуту.**
Роздрібна та оптова закупівля продукту.	Збут, оновлення та налаштування товару.	Усі можливі канали збуту.	Власна.

Висновки

Отже даний стартап(розробка модифікованого криптографічного методу)може привласнити клієнтів конкурентів за рахунок кращої

якості,швидкодії та нових можливостей приладу. Цей висновок можна зробити за результатами маркетингових досліджень. Вже починаючи продажі приладу,є прогноз по високим доходам. Прилад може бути затребуваним серед компаній,що цінують конфеденційність, для людей, які бажають тримати у секреті власні дані. Прямими конкурентами для даного стартапу є:

1. Криптопровайдер ViPNet CSP 4 **.**
2. **“**КриптоАРМ**”** - универсальна програма для шифрування і електронного підпису файлів.

Непрямими конкурентами є інтернет форуми.

Порядок дій при виході на ринок визначається у наступних діях:

1. Реклами власного винаходу,за рахунок створення сайту(піар).
2. Знаходження коштів на першу партію приладів від зацікавлених клієнтів.
3. Створення першої хвилі приладів і для сбору позитивних коментарів та подальших рекомендацій товару покупцями своїм друзям(за рахунок продаж по низький ціні).
4. На виручені кошти створити наступну партію товару,яка вже буде купуватися користувачами,які побачили або почули позитивні відгуки.

Клієни вкладуться у першу партію приладів,що погасить половину трат, другу половину-інвестори. При кожній наступній партії створення приладів,можна збільшувати ціну,щоб досягти переходу на дохід тільки від продаж та не використовувати інвесторські кошти.

4. КОНСТРУКТОРСЬКО-ТЕХНОЛОГІЧНА ЧАСТИНА

4.1 Робота пристрою

При запуску пристрою насамперед відбувається первісна ініціалізація всіх необхідних для функціонування пристроїв. Після даної операції програмне забезпечення починає працювати в нескінченному циклі. У циклі послідовно виконуються следующе операції:

Відображення запиту на введення ключа iButton.

Читання встановленого ключа.

Перевірка ідентифікатора ключа на наявність відповідного йому запису в базі, що зберігається на SD-карті. Якщо ідентифікатор був при вході, тоді виконується повторне запрограмування телефону запитувати код ключа iButtonна екран дисплея.

Генерація і вивід на дисплей клавіатури введення ПІН-коду з випадковим розташуванням клавіш.

Зчитування введеного користувачем ПІН-коду.

Перевірка введеного ПІН-коду на відповідність введеному раніше ключу iButtonпо базі даних на SD-карті.

Відображення зображення, що відповідає результату перевірки коректності виведеного ПІН-коду.

Затримка перед подальшим виведенням нового запиту на введення ключа iButton.

Запишемо етапи роботи пристрою:

1 етап

На даному етапі аналогова шумова послідовність перетворюється АЦП для отримання двійкової випадково послідовності. Для подальшої роботи будуть використовуватися кілька молодших біт, отриманих з АЦП.

2 етап

Виходять перші 22 біта, значення яких повинно бути менше 10 . Якщо отримане значення більше, тоді дана послідовність відкидається і аналізуються наступні 22 біта. Як тільки умова виконана, необхідно відновити розташування кнопок на дисплеї. Для цього буде використана

функція PERMLEXUNRANK (n, r): Ця функція не дозволяє однозначно відновити шукану послідовність значень на екрані дисплея.

Опис модулів програмного забезпечення

У цій частині буде проведений короткий огляд кожного з модулів програмного забезпечення. Більш детальна інформація по кожному з модулів наведена в електронній документації до розробленого коду.

Низькорівневий і системний код

Модуль «System» містить низькорівневий системний код, в тому числі функцію ініціалізації мікроконтролера LPC2148 і обробники переривань.

Драйвер роботи із затримками

Драйвер призначений для виконання функцій формування затримок заданої величини і вимірювання часових інтервалів. У своїй роботі драйвер використовує блок Timer1 мікроконтролера LPC2148.

Призначення функції: виконує ініціалізацію драйвера роботи з затримками, в тому числі налаштування параметрів блоку Timer1 мікроконтролера.

Аргументи функції: відсутні.

Призначення функції: Виконує ініціалізацію модуля роботи з контролером DS2482-100, включаючи ініціалізацію модуля роботи з шиною I2C.

Аргументи функції: відсутні.

Призначення функції: Здійснює читання одного байта даних від контролера DS2482-100.

Призначення функції: Здійснює відправку команди без параметрів контролера DS2482-100 +.

Виконує ініціалізацію модуля роботи з ідентифікаторами iButton, включаючи ініціалізацію модулів роботи з контролером DS2482-100 і з шиною I2C.

Аргументи функції: відсутні.

Значення, що повертається: __TRUE при успішному завершенні ініціалізації; _FALSE в іншому випадку.

4.2 Складові частини пристрою

Драйвер роботи з картами пам'яті формату Secure Digital (SD)

Драйвер забезпечує роботу з картами пам'яті формату Secure Digital (SD). Включає модулі роботи з шиною SPI і безпосередньо картами пам'яті. Використовує блок SPI мікроконтролера.

До складу драйвера роботи з картами пам'яті формату Secure Digital входять наступні файли:

«Mmc.c» - код драйвера роботи з картами пам'яті форматаSecure Digital (SD);

«Mmc.h» - заголовки драйвера роботи з картами пам'яті формату Secure Digital (SD).

функція mmc_deselect

Сигнатура функції: void mmc_deselect (void).

Призначення функції: Завершує сеанс обміну інформацією з SD-картою.

Аргументи функції: відсутні.

Значення, що повертається: відсутній.

функція mmc_init

Сигнатура функції: BOOL mmc_init (void).

Призначення функції: Виконує ініціалізацію модуля роботи з картами пам'яті, включаючи ініціалізацію модуля роботи з шиною SPI.

Аргументи функції: відсутні.

Значення, що повертається: __ TRUE при успішної ініціалізації; __FALSE в іншому випадку.

До складу драйвера роботи з графічним дисплеєм входять наступні файли:

«Tft.c» - код драйвера роботи з графічним дисплеєм;

«Tft.h» - заголовки драйвера роботи з графічним дисплеєм.

функція ssp_busspeed

Сигнатура функції: BOOL ssp_busspeed (U32 kbaud).

Призначення функції: Встановлює задану швидкість обміну по інтерфейсу SPI для блоку SSP мікроконтролера.

Аргументи функції: kbaud [вхід] швидкість обміну в кілободах.

Значення, що повертається: __TRUE при успішному завершенні операції; __FALSE в іншому випадку.

функція ssp_byte

Сигнатура функції: U8 ssp_byte (U8 outb).

Призначення функції: Виконує одночасно передачу і прийом одного байта по шині SPI.

Аргументи функції: outb [Вхід] передається байт.

Значення, що повертається: Значення, отримане по шині SPI (1 байт).

функція ssp_init

Сигнатура функції: BOOL ssp_init (void).

Призначення функції: Виконує ініціалізацію модуля роботи з шиною SPI, в тому числі налаштування параметрів блоку SSP мікроконтролера.

Аргументи функції: відсутні.

Значення, що повертається: __TRUE при успішної ініціалізації; __FALSE в іншому випадку.

функція ssp_setcs

Сигнатура функції: BOOL ssp_setcs (U32 ss).

Призначення функції: Перекладає лінію CS (SSEL) інтерфейсу SPI в заданий стан.

Аргументи функції: ss [вхід] необхідний стан лінії (0 - низький рівень, інше значення - високий рівень).

Значення, що повертається: __TRUE при успішному завершенні операції; __FALSE в іншому випадку.

функція ssp_use16

Сигнатура функції: BOOL ssp_use16 (BOOL use16).

Призначення функції: Здійснює вибір розрядності однієї посилки по шині SPI: 8 або 16 біт.

Аргументи функції: use16 [вхід] ознака використання 16-розрядних посилок.

Значення, що повертається: __TRUE при успішному завершенні операції; __FALSE в іншому випадку.

функція ssp_word

Сигнатура функції: U16 ssp_word (U16 outw).

Призначення функції: Виконує одночасно передачу і прийом одного слова (2 байта) по шині SPI.

Аргументи функції: outw [вхід] передане слово (2 байта).

Значення, що повертається: Значення, отримане по шині SPI (2 байта).

функція tft_bitmap

Сигнатура функції: void tft_bitmap (BITMAP * bmp, U16 * palette, U16 x, U16 y).

Призначення функції: Здійснює висновок растрового зображення на дисплей з додатковим палітровим перетворенням квітів.

Аргументи функції:

bmp [вхід] структура, що описує растрове зображення;

palette [вхід] палітра, яка містить кольори в форматі R5G6B5;

x [вхід] X-координата лівого верхнього кута області виведення;

y [вхід] Y-координата лівого верхнього кута області виведення.

Значення, що повертається: відсутній.

функція tft_bitmap_raw

Сигнатура функції: void tft_bitmap_raw (U16 * bmp, U16 x, U16 y, U16 w, U16 h).

Призначення функції: Здійснює висновок растрового зображення на графічний дисплей без додаткової обробки.

Аргументи функції:

bmp [вхід] растрове зображення, записане по рядках в колірному форматі R5G6B5.

x [вхід] X-координата лівого верхнього кута області виведення;

y [вхід] Y-координата лівого верхнього кута області виведення;

w [вхід] ширина області;

h [вхід] висота області.

Значення, що повертається: відсутній

функція tft_clear

Сигнатура функції: void tft_clear (U16 color).

Призначення функції: Виконує заливку всій області графічного дисплея заданим кольором.

Аргументи функції: color [вхід] колір заливки в форматі R5G6B5 (2 байта).

Значення, що повертається:

функція tft_init

Сигнатура функції: BOOL tft_init (void).

Призначення функції: Виконує ініціалізацію модуля роботи з графічним дисплеєм, включаючи ініціалізацію модуля роботи з інтерфейсом SPI.

Аргументи функції: відсутні.

Значення, що повертається: __TRUE при успішної ініціалізації; __FALSE в іншому випадку.

функція tft_pixel

Сигнатура функції: void tft_pixel (U16 x, U16 y, U16 color).

Призначення функції: Здійснює висновок точки заданого кольору на графічний дисплей.

Аргументи функції:

x [вхід] X-координата точки;

y [вхід] Y-координата точки;

color [вхід] колір точки в форматі R5G6B5 (2 байта).

Значення, що повертається: відсутній.

функція tft_rectangle

Сигнатура функції: void tft_rectangle (U16 x1, U16 y1, U16 x2, U16 y2, U16 color).

Призначення функції: Здійснює заливку прямокутної області графічного дисплея заданим кольором.

Аргументи функції:

x1 [вхід] X-координата лівого верхнього кута області;

y1 [вхід] Y-координата лівого верхнього кута області;

x2 [вхід] X-координата правого нижнього кута області;

y2 [вхід] Y-координата правого нижнього кута області;

color [вхід] колір заливки в форматі R5G6B5 (2 байта).

Значення, що повертається: відсутній.

функція tft_reset

Сигнатура функції: void tft_reset (void).

Призначення функції: Відновлення наведених контролера графічного дисплея.

Аргументи функції: отсутсвтие.

Значення, що повертається: відсутній.

функція tft_window

Сигнатура функції: void tft_window (U16 x, U16 y, U16 w, U16 h)

Призначення функції: Встановлює активну область графічного дисплея.

Аргументи функції:

x [вхід] X-координата лівого верхнього кута області;

y [вхід] Y-координата лівого верхнього кута області;

w [вхід] ширина області;

h [вхід] висота області.

Значення, що повертається: відсутній.

функція tft_write_cmd

Сигнатура функції: void tft_write_cmd (U8 cmd).

Призначення функції: Виконує передачу команди контролеру графічного дисплея.

Аргументи функції: cmd [вхід] код команди (1 байт).

Значення, що повертається: відсутній.

функція tft_write_dat

Сігнатурафункціі: void tft_write_dat (U16 dat).

Призначення функції: Виконує передачу даних контролера графічного дисплея.

Драйвер роботи з сенсорним екраном

Драйвер роботи з сенсорним екраном забезпечує функціонування сенсорного екрану і використовує канали 3 і 4 аналого-цифрового перетворювача AD1 мікроконтролера.

До складу драйвера роботи з сенсорним екраном входять наступні файли:

«Touch.c» - код драйвер роботи з сенсорним екраном;

«Touch.h» - заголовки драйвера роботи з сенсорним екраном

функція read_ad1_3

Сігнатурафункціі: U16 read_ad1_3 (void).

Призначення функції: виконує читання показань з каналу 3 аналого-цифрового перетворювача AD1.

Аргументи функції: відсутні.

Значення, що повертається: значення в інтервалі від 0 до 1023, що відповідає напрузі на вході каналу 3 аналого-цифрового перетворювача AD1.

функція read_ad1_4

Сігнатура функціі: U16 read_ad1_4 (void).

Призначення функції: виконує читання показань з каналу 4 аналого-цифрового перетворювача AD1.

Аргументи функції: відсутні.

Значення, що повертається: значення в інтервалі від 0 до 1023, що відповідає напрузі на вході каналу 4 аналого-цифрового перетворювача AD1.

функція touch_calibrate

Призначення функції: виконує калібрування сенсорного екрану.

Аргументи функції:

width [вхід] ширина логічного екрану (як правило, дозвіл графічного дисплея по осі X в пікселах);

height [вхід] висота логічного екрану (як правило, дозвіл графічного дисплея по осі Y в пікселах);

touch_x_left [вхід] показання сенсорного екрану по осі X, відповідні дотику до лівого краю екрана;

touch_x_right [вхід] показання сенсорного екрану по осі X, відповідні дотику до правого краю екрану;

touch_y_top [вхід] показання сенсорного екрану по осі Y, відповідні дотику до верхнього краю екрану;

touch_y_bottom [вхід] показання сенсорного екрану по осі Y, відповідні дотику до нижнього краю екрану;

threshold [вхід] порогове значення, яке використовується при визначенні дотику до сенсорного екрану.

Значення, що повертається: відсутній.

функція touch_detect

Сігнатурафункціі: BOOL touch_detect (void).

Призначення функції: визначає факт дотику до сенсорного екрану за допомогою читання показань по осі X. Дотик вважається зафіксованим, якщо показання нижче порогового значення, що задається змінною touch_threshold.

Аргументи функції: відсутні.

Значення, що повертається: _TRUE, якщо зафіксовано дотик до сенсорного екрану; __FALSE в іншому випадку.

функція touch_idle

Сигнатура функції: void touch_idle (void).

Призначення функції: Перекладає висновки мікроконтролера, підключені до сенсорного екрану, в початковий стан. Може використовуватися для відключення сенсорного екрану в періоди його неактивності.

Аргументи функції: відсутні.

Значення, що повертається: відсутній.

функція touch_init

Сігнатурафункціі: BOOL touch_init (void).

Призначення функції: виконує ініціалізацію модуля роботи з сенсорним екраном, в тому числі налаштування параметрів роботи аналого-цифрового перетворювача AD1.

Драйвер роботи з універсальним асинхронним приймачем-передавачем використовує блок UART0 мікроконтролера і забезпечує роботу налагоджувального порту.

4.3 Принципова схема пристрою

Принципова схема представлена на рисунку 4.1.

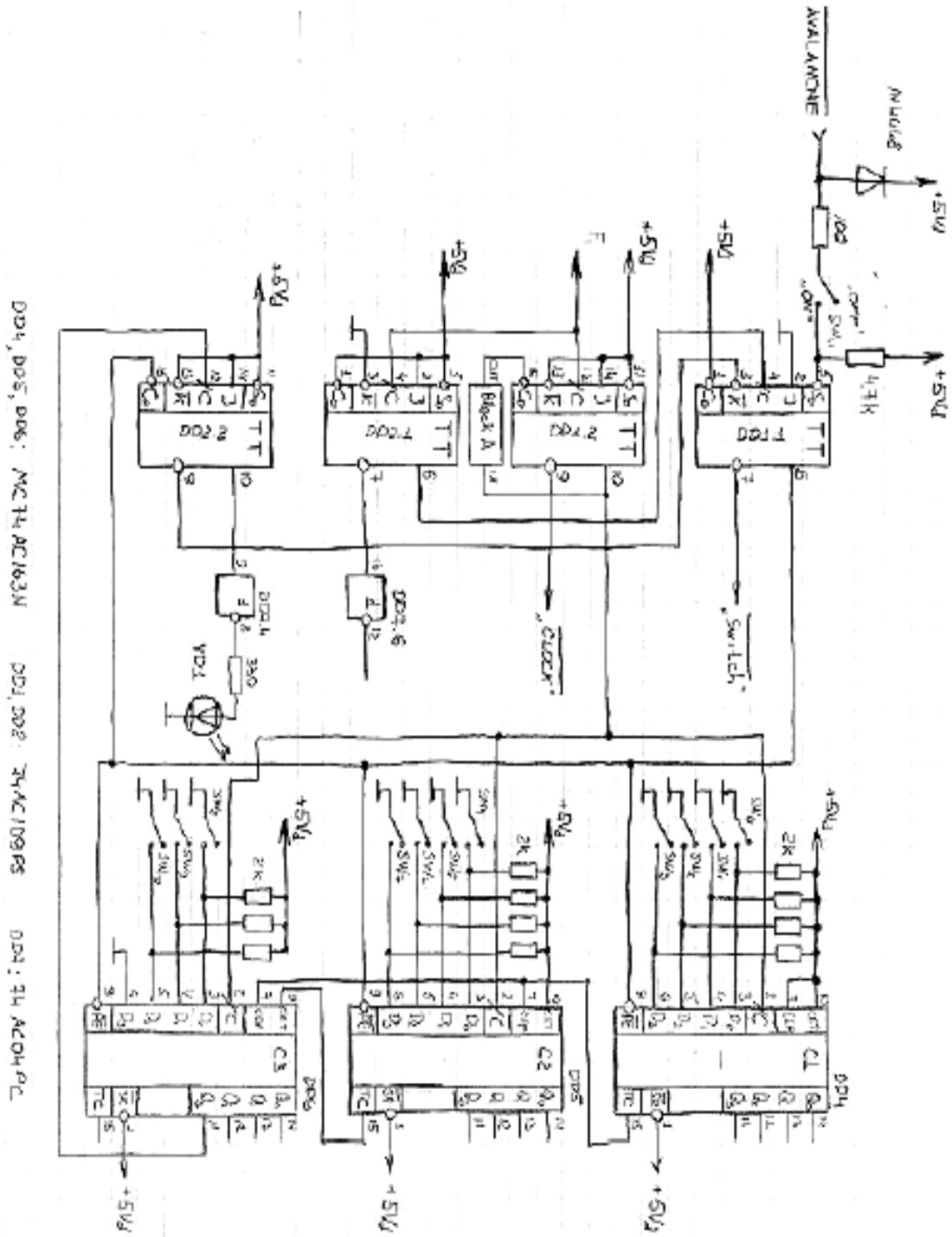


Рисунок 4.1 – Принципова схема

4.4 Лістинг програми пристрою

Наведемо лістинг програми розглянутого пристрою:

```
using System;

using System.Linq;

using System.Text;

using System.IO;


namespace PRSlength
{
    public class PRSlength
    {

        public PRSlength()
        {
            init();
        }


        private void init()
        {

        }

        public void calculate (int onePRSlength, int numofS, string filename)
        {
            FileStream fs;
            fs = new FileStream (filename, FileMode.Open, FileAccess.Read);

            //int period = 1; //calculating period (in elements) for given n (or cBitblock)
            //for (int i = (int)Math.Pow ((double)2,(double)cBitblock); i!= 0; i--) {
            //    period *= i;
```

```
//}
//period *= (int)Math.Pow ((double)2, (double)(2 * cBitblock));

//int tlenreal = period*cBitblock; //calculating period (in bits)

int tlen = onePRSlength;

if (onePRSlength > 100000000) {
    tlen = 100000000;
}

int [] allpers = new int[numofS]; //stores one period per sequence (all periods of all seqs)
int [] colofpers = new int[numofS]; // stores the amount of each period
int [] predper = new int[numofS]; //stores pre-periods, if exist, for each PRS that has them

Console.WriteLine("Sequences total: "+numofS);



//0. Cycle for seeking periods in all sequences
for (int r = 0; r < numofS; r++) {

    int ctr = 0;
    bool equal = false;
    allpers[r] = 0;
    predper[r] = 0;

    bool [] Bo = new bool[tlen]; // an array where bits of PRS will be stored

    //1.  Reading bytes from file to array-----
    for (int j = 0; j<tlen/8; j++) {

        byte b = (byte)fs.ReadByte ();

        for (int t = 128; t >0; t = t/2) {
```

```
            if ((b & t) != 0) {
                Bo[ctr] = true;
            }
            if ((b & t) == 0) {
                Bo[ctr] = false;
            }
            //Console.Write (Bo[ctr]+" ");
            ctr++;
        }
    }//1. end-----
    Console.WriteLine("{0}. bytes read;", r);



    //2.1  Calculating period-----
    for (int i = 1; i<tlen; i++) {
        //Console.Write("i = "+i);
        for (int k = 1; k<=Math.Truncate((double)(tlen/i)); k++){
            //Console.WriteLine(" k = "+k);
            for (int j = 0; j<i; j++) {

                if (j + i*k >= tlen)
                    break;
                if (Bo [j] == Bo [j + i*k]) {
                    equal = true;
                } else {
                    equal = false;
                    break;
                }

            }
            if (equal == false) break;
        }
        if (equal == true) {
            allpers[r] = i;
```

```
                break;
            }
        }//2.1  end-----

        //2.2 If no period found, seek for period with predperiod
        if (allpers[r] == 0)
        {
            for (int i = 1; i<tlen; i++) { //each i is an alleged period
                //Console.Write("i = "+i);
                for (int shift = 0; shift<tlen; shift++) // cutting first 'shift' bits of PRS and seeking
again
                {
                    for (int k = 1; k<=Math.Truncate((double)(tlen/i)); k++){ //k = PRS length / i
                        //Console.WriteLine(" k = "+k);
                        for (int j = 0; j<i; j++) { //j is the number of compared blocks in alleged i-
period

                            if (shift+j + i*k >= tlen)
                                break;
                            if (Bo [shift+j] == Bo [shift+j + i*k]) {
                                equal = true;
                            } else {
                                equal = false;
                                break; //break from j-cycle
                            }

                        }
                        if (equal == false) break; //break from k-cycle and try next shift
                    }
                    if (equal == true) {
                        allpers[r] = i;
                        predper[r] = shift;
                        break;} // break from shift-cycle
                }
                if (equal == true) {break;} //break from i-cycle
```

```
        }//2.2  end-----
    }

}//0. end-----

fs.Close ();





//4.   Counting the amount of each period-----
bool uniq = false;
for (int i = 0; i< numofS; i++) {
    if (i == 0) {uniq = true;}
    colofpers[i] = 0;
    for (int j = 0; j<i; j++) {
        if (allpers[i] == allpers [j]) {uniq = false; break;}
        else {uniq = true;}
    }
    if (uniq == false) continue;
    else{
        for (int j = i; j<numofS; j++)
        {
            if (allpers[i] == allpers[j]) colofpers[i]++;
        }
    }
}//4. end------

//6.  Displaying and exporting the results.-----
StreamWriter sw = new StreamWriter ("results.txt");
sw.WriteLine ("RESULTS FOR SEEKING PERIODS OF PRS.\n");
sw.WriteLine ("Sequences read in total: {0}", numofS);
//sw.WriteLine ("Element consists of {0} bits (n = {0}).\n", cBitblock);
```

```
            sw.WriteLine ("PERIODS:");
            int perctr = 1;
            for (int i = 0; i<numofS; i++) {
               if (colofpers[i] == 0) continue;
               else{
                  Console.Write("{0}. {1} sequences have period = {2} bits;", perctr, colofpers[i], all
pers[i]);
                  sw.Write("{0}. {1} sequences have period = {2} bits;", perctr, colofpers[i], allpers[i
]);
                  perctr++;

                  if (predper[i] !=0) {
                     Console.WriteLine(" Pre-period = {0};", predper[i]);
                     sw.WriteLine(" Pre-period = {0};", predper[i]);
                  }
                  else {Console.WriteLine();
                     sw.WriteLine();}
               }
            }//6.  end-----

            sw.Close ();

         }

      }

      class MainClass
      {
         public static void Main (string[] args)
         {
            Console.Write ("\n\nHello! This program seeks a period (if exists) of a pseudo-
random\n" +
                        "sequence (PRS) of 0s and 1s\n");

            PRSlength prs = new PRSlength ();
```

```
int conePRSlength, cNumofS;
string sconePRSlength, scNumofS;

Console.Write ("\nEnter the name of the input file: ");
string filename = Console.ReadLine ();

do {Console.Write ("\nEnter the length of a single PRS in bits: ");
   sconePRSlength = Console.ReadLine ();
} while (!int.TryParse (sconePRSlength, out conePRSlength) | conePRSlength <0);

do {Console.Write ("\nHow many PRSs are there in the input file?: \n");
   scNumofS = Console.ReadLine ();
} while (!int.TryParse (scNumofS, out cNumofS));

prs.calculate (conePRSlength, cNumofS, filename);

Console.WriteLine ("\nDONE!!! The results were written to results.txt in the same direct
ory.")
```

Висновки до розділу 4

В даному розділі було розроблено принципову схему криптографічного шифрувального пристрою. Описана покрокова робота пристрою та проведено дослідження складових частин пристрою та функцій, які вони виконують. Наведений лістинг програми розглянутого пристрою.

РОЗДІЛ 5. ЕКСПЕРИМЕНТАЛЬНА ЧАСТИНА

5.1 Структурна схема експериментальної установки

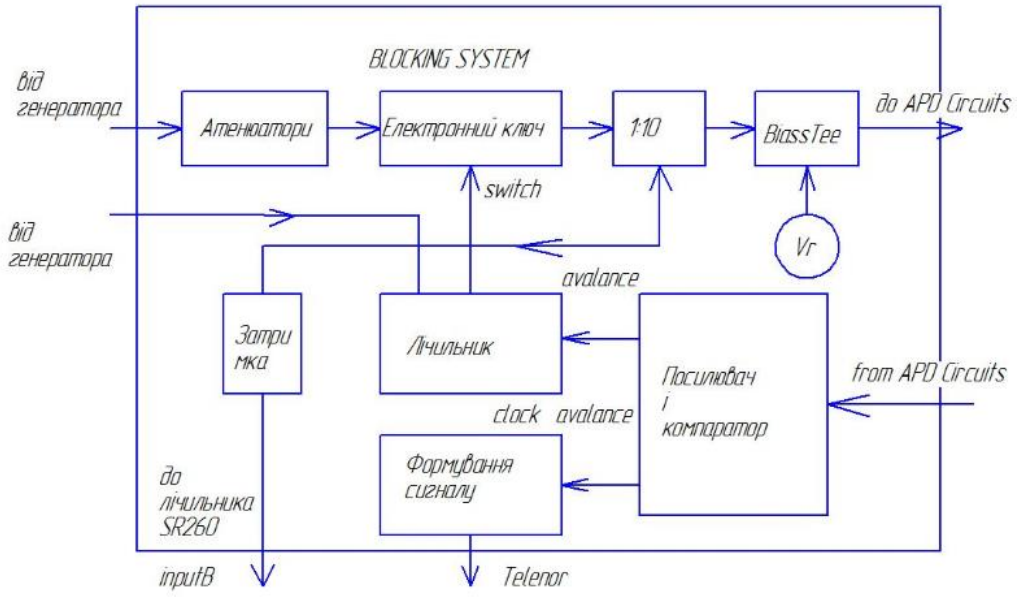


Рисунок 5.1 – Структурна схема експериментальної установки

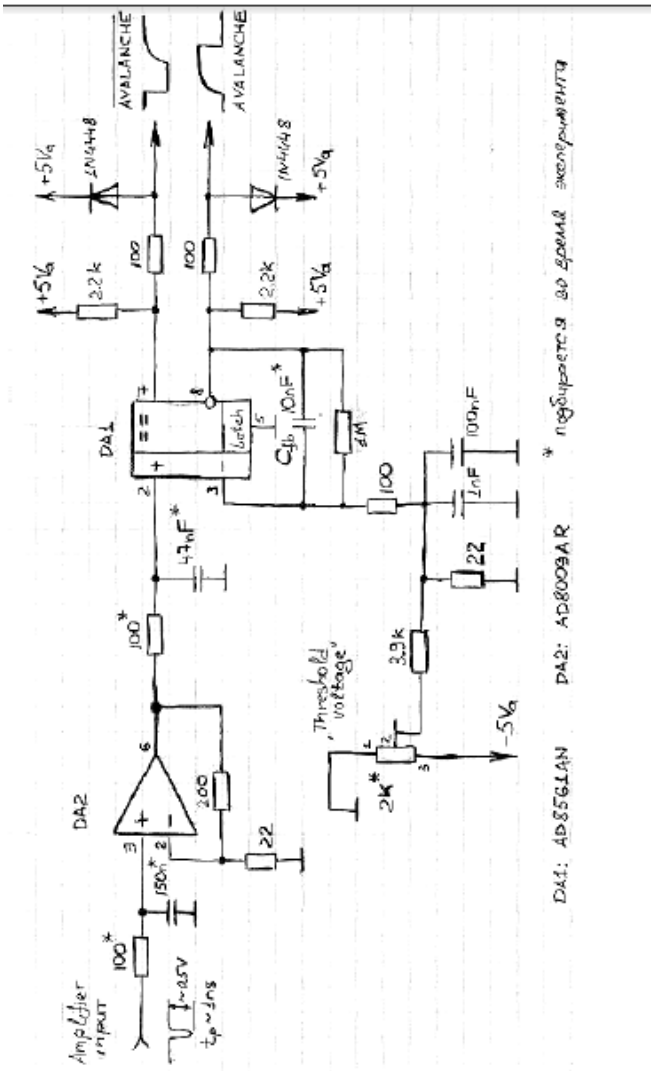


Рисунок 5.2 – Схема підсилювача

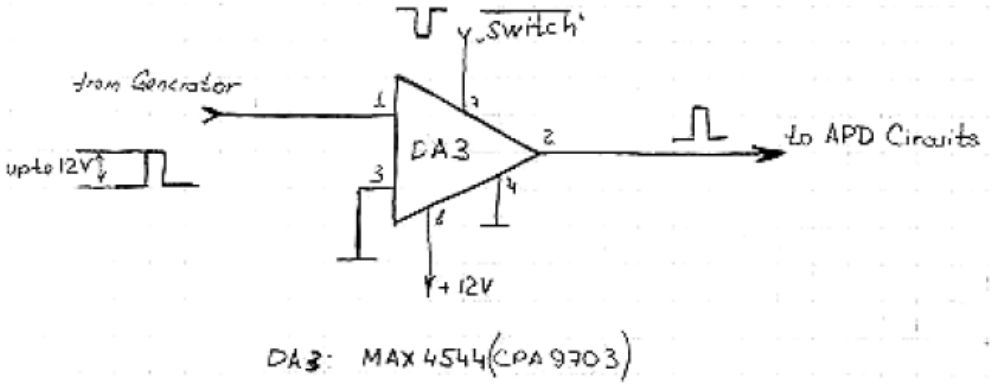


Рисунок 5.3 – Схема підключення електронного ключа

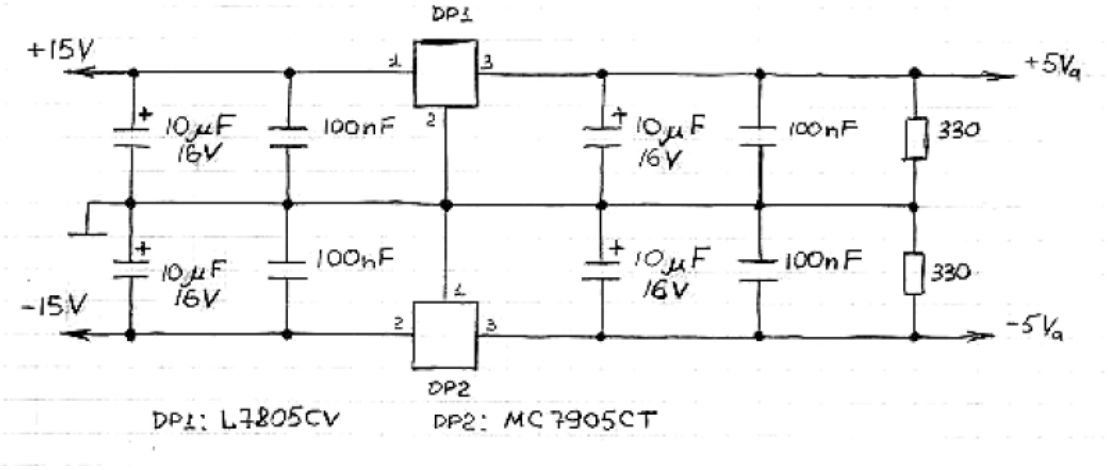


Рисунок 5.4 – Схема блоку живлення

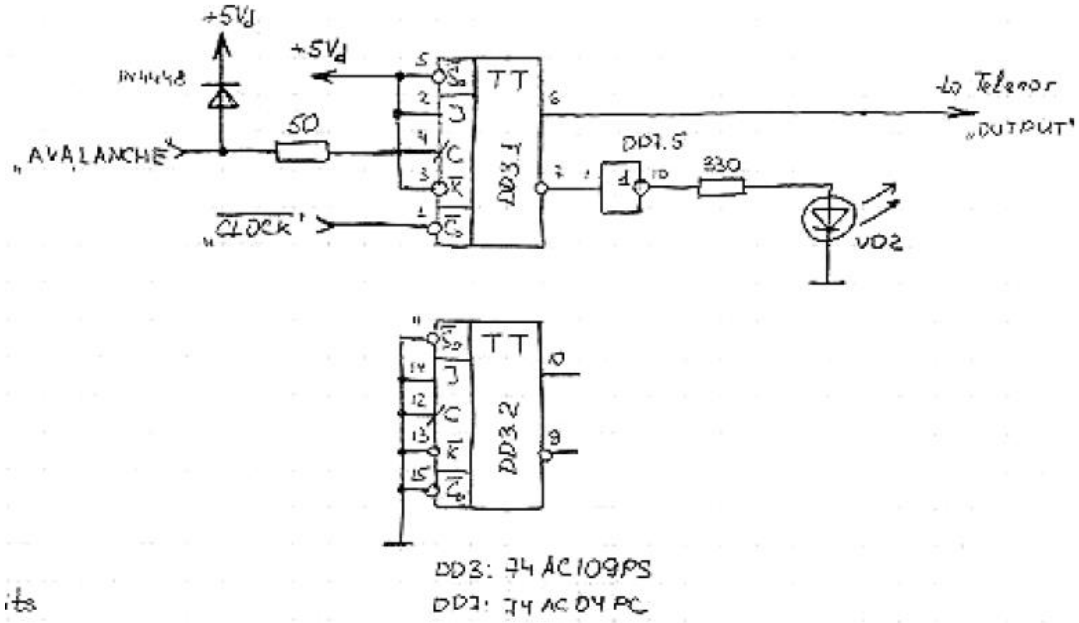


Рисунок 5.5 – Схема формувача

5.2 Лістинг модифікованої програми

Лістинг програми для модифікації наведений нижче.

```
using System;
using System.Linq;
using System.Text;
using System.IO;
namespace RC4
{
   public class RC4
```

```
{
    int x = 0;
    int y = 0;
    int dim;
    int bitblock;
    byte[] S = new byte[256]; //all permutations (= 2^bitblock = dim) are stored here.

    string nistfilename = "bit5prs.txt";

    int PRSlength;
    int numofS;
    int seed;

    public RC4(int cBitblock, int cDim, int cPRSlength, int cNumofS, int cSeed)
    {
        init(cBitblock, cDim, cPRSlength, cNumofS, cSeed);
    }

    // Initialization of Class
    private void init(int cBitblock, int cDim, int cPRSlength, int cNumofS, int cSeed)
    {
        bitblock = cBitblock;
        dim = cDim;
        PRSlength = cPRSlength;
        numofS = cNumofS;
        seed = cSeed;
        Console.WriteLine ("\nbit-block = {0}", bitblock);
        Console.WriteLine ("number of permutations = {0}", dim);
        Console.WriteLine ("PRS length = {0}", PRSlength);
        Console.WriteLine ("number of sequences = {0}", numofS);
        Console.WriteLine ("seed = {0}", seed);
    }



    // Pseudo-Random Generation Algorithm
```

```
private byte randBlock()
{
    x = (x + 1) % dim;
    y = (y + S[x]) % dim;

    S.Swap(x, y);

    /*for (int w = 0; w<dim; w++) {
        Console.Write (S [w]);
    }

    Console.Write(", x = {0}, y = {1} ", x, y);
    //Console.WriteLine();
    */
    return S[(S[x] + S[y]) % dim];
}


//Generating PRS and writing it to files
public void toFile ()
{
    byte[] randbits = new byte[8];  //bitblocks, generated by every version of RC4. Ex: bits in
 block: 6, 6*8 = 48 bits
    byte[] bit8 = new byte[bitblock]; //finished bytes of PRS, the result of distributing of bitbl
ocks (8*6 = 48 bits)
    ulong nabor; // a bit stack, a set of bitblocks, that are then distributed to bytes

    FileStream fout = null;
    fout = new FileStream (nistfilename, FileMode.OpenOrCreate); // file, containing bytes o
f PRS for NIST testing

    int bytecounter = 0; //counts bytes in PRS

    Random rnd = new Random(seed);
    byte [,] Temp = new byte[numofS, dim]; //array of S-arrays
```

```
//Making ready-for-use S-arrays---//
if (bitblock == 2) {
    int c = 0;
    numofS = 24;
    for (int i = 0; i<4; i++) {
        for (int j = 0; j<4; j++) {
            for (int k = 0; k<4; k++) {
                for (int s =0; s<4; s++) {
                    if (i != j & (k != i & k != j) & (s != i & s != j & s != k)) {
                        Temp [c, 0] = (byte)i;
                        Console.Write (Temp [c, 0]);
                        Temp [c, 1] = (byte)j;
                        Console.Write (Temp [c, 1]);
                        Temp [c, 2] = (byte)k;
                        Console.Write (Temp [c, 2]);
                        Temp [c, 3] = (byte)s;
                        Console.Write (Temp [c, 3]);
                        c++;
                        Console.WriteLine ();
                    }//if (c == numofS) break;
                }
            }
        }
    }
}

else {
    for (int s =0; s< numofS; s++) {
        bool equal = false;
        int eq = 0;

        for (int i = 0; i < dim; i++) { //initializes Temp S-arrays
```

```
    Temp [s, i] = (byte)i;
}
for (int i = 0; i < 2*dim; i++) { //swaps random permutations in current S-array
    Temp.Swap2 (s, rnd.Next (dim), rnd.Next (dim));
}

//checking if the last S-array is equal to some previous
for(int i = 0; i<s; i++)
{
    for (int j = 0; j<dim; j++)
    {
        if(Temp [s, j] != Temp [i, j]) {equal = false; break;}
        else {equal = true;}
    }
    if (equal == true) {eq = i; break;}

}

//making this S-array unique
while (equal == true)
{
    Console.WriteLine("permutation {0} got like {1}", s, eq);
    for (int i = 0; i < 2*dim; i++) { //swaps random permutations in current S-array
        Temp.Swap2 (s, rnd.Next (dim), rnd.Next (dim));
    }

    for(int i = 0; i<s; i++)
    {
        for (int j = 0; j<dim; j++)
        {
            if(Temp [s, j] != Temp [i, j]) {equal = false; break;}
            else {equal = true;}
        }
        if (equal == true) break;
    }
```

```
            //Console.WriteLine("permutation {0}: equal = {1}", s, equal);
        }//

    }
}//end---//
```

Висновки до розділу 5

Цей розділ включає в себе розроблену структурна схема експериментальної установки та представлена схема підсилювача. Створена программа для модифікації, лістинг якої наведений у розділі. Даний прилад із розробленою програмою відповідають вимогам захисту інформації з використанням квантової криптографії.

ВИСНОВКИ

Основним завданням криптографії було тільки шифрування повідомлень— перетворенням зрозумілої форми в зашифровану та у зворотньому напрямку(у одержувача),і розшифрувати дане повідомлення без секретного ключа,який використовується для того щоб дешифрувати дане повідомлення,не можна. Але наразі криптографія відповідає не лише за шифрування повідомлень,але вона і включає в себе перевірку цілістонсті переданого повідомлення,функції ідентифікації та аутентифікації,електронні підписи тощо.

Наразі впроваджено різноманітну кількість технологій захисту даних ,різниця між якими полягає і в ступенях надійності,і в практичному впровадженні,і в принципах,що покладені в основу даних методів. Класична криптографія продовжує свій розвиток в криптографії, основа якої є теорія ґраток та криптосистемах, що базуються на синдромах. Метою є пошук методів квантової та постквантової криптографії, визначення їхніх переваг та недоліків, а також перспектив практичної реалізації.

Квантовий розподіл ключів-це єдиний метод, де впроваджують комерційні рішення. Ці рішення можна використовувати для симетричного шифрування, та квантовий прямий безпечний зв'язок, що реалізується в лабораторіях, що свідчить про можливе створення у майбутньому комерційних рішень.

Наразі теоретичні моменти безпеки квантової криптографії є дуже інтенсивним джерелом досліджень, що постійно розвивається,заважають їх швидкому розвитку лише технологічні проблеми, які у найближчому майбутньому можуть бути вирішені.

Розвиток постквантових систем відбувається дуже повільно, бо одна із головних на те причин: велика різниця між «крипостійкі теоретично» і

«крипостійкі практично» між системами асиметричного шифрування на основі задач теорії ґраток.Але це також тільки дає можливість проводити дослідження у перспективному маловивченому напрямку фундаментальних і прикладних математичних досліджень в галузі постквантової криптографії.

Список наукової літератури

1. Fluhrer S.R. Statistical analysis of the alleged rc4 keystream generator. Proceedings of the 7th International Workshop on Fast Software Encryption/ S.R. Fluhrer, D.A. McGrew. - 2000.

2. Mantin I.A. A practical attack on broadcast RC4. Revised Papers from the 8th International Workshop on Fast Software Encryption / I.A. Mantin, A. K. Shamir.-Fse'01.-2001.- P.152-164.

3. Grosul A. L. A related-key cryptanalysis of RC4/ A.L. Grosul, D.S. Wallah.-Technical Report TR-00-358, Department of Computer Science, Rice University.-2000.

4. Fluhrer S.R. Weaknesses in the Key Scheduling Algorithm of RC4/ S.R. Fluhrer, I.A. Mantin, A.K. Shamir.- Revised Papers from the 8th Annual International Workshop on Selected Areas in Cryptography.-2001.

5. Souradyuti P.L. Analysis of Non-fortuitous Predictive States of the RC4 Keystream Generator/ P.L. Souradyuti, B.T. Preneel.- INDOCRYPT.- 2003.

6. Kim M.O. Investigation about the RC4 and the weakness of WEP/ M.O. Kim // Data Security & Cryptography.- 2004.

7. Ohigashi T.H. New Weakness in the Key-Scheduling Algorithm of RC4/ T.H. Ohigashi, Y.B. Shiraishi, M.C. Morii//IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences.-2008.

8. Шнайер Б. Прикладная криптография. Протоколы, алгоритмы, исходные тексты на языке Си/ Б. Шнайер.-М.:Триумф, Москва.- 2002.-610с.

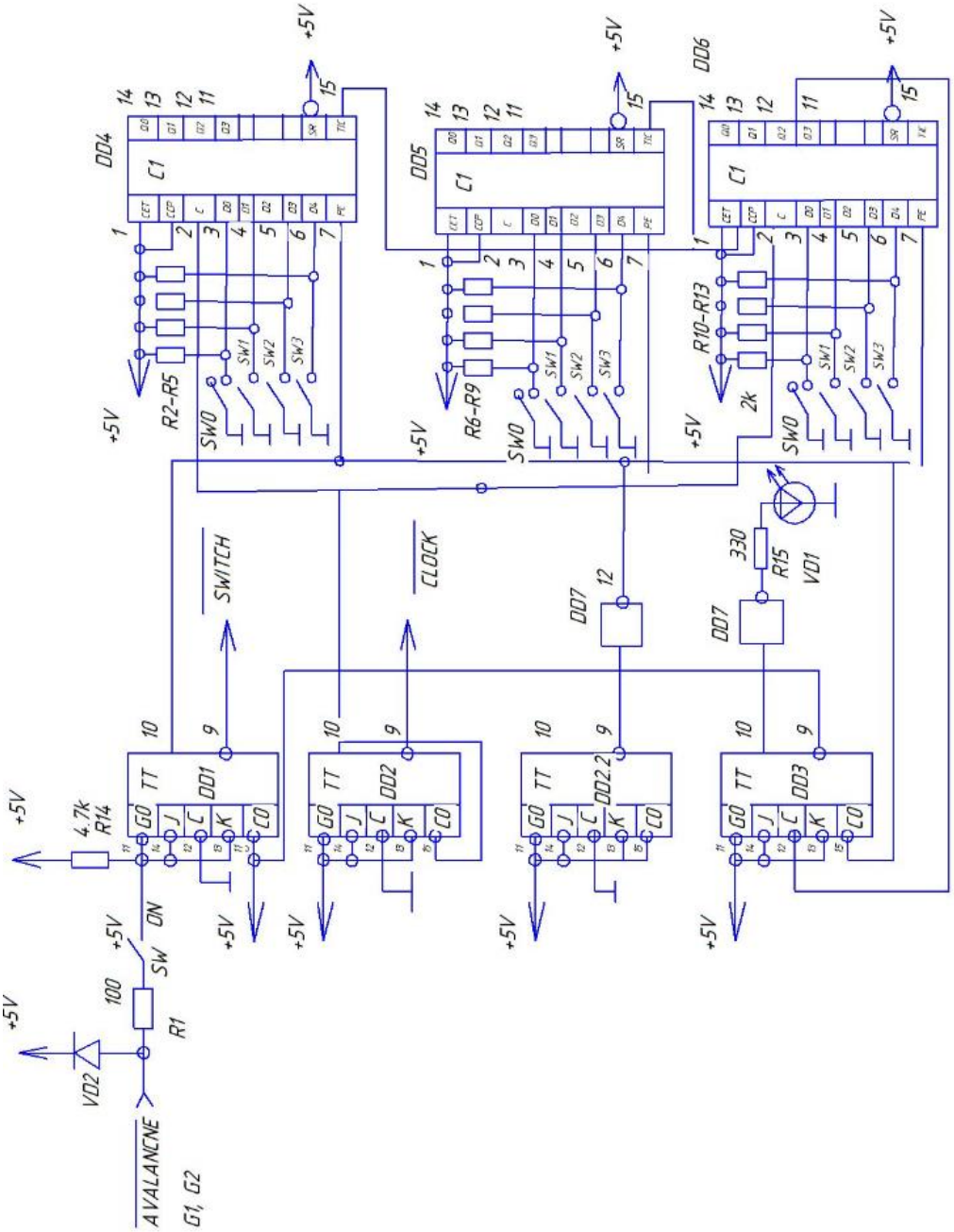
9. Кнут Д.Э. Искусство программирования/ Д.Э. Кнут.- М.:Вильямс,Москва.- 3 изд, 2 том.- 2005.

ДОДАТКИ

ДОДАТОК 1 ПЕРЕЛІК ЕЛЕМЕНТІВ

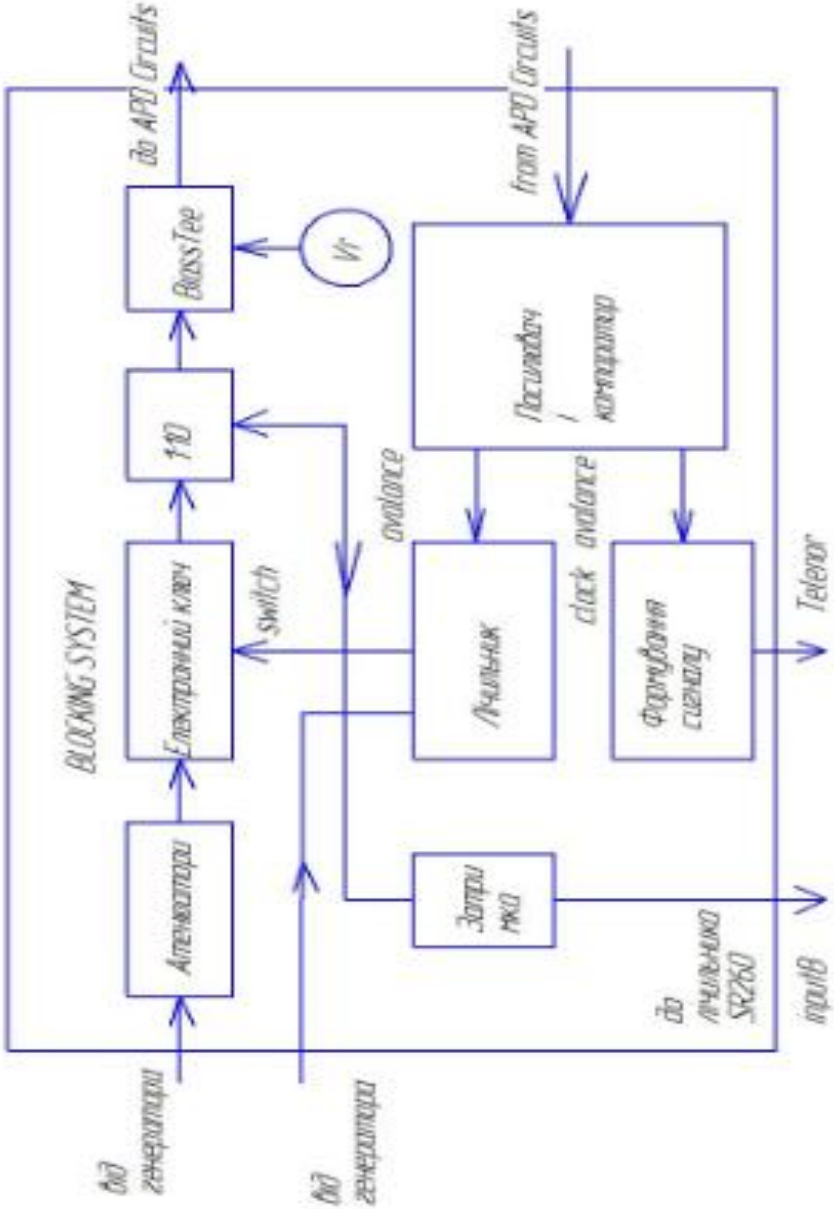
По. Обозн.	*Найменування*	*Кільк.*	*Примітка*
Елем.	*Діоди*	*Кільк.*	*Примітка*
VD1	1N4007	*1*	
VD2	КВ104А	*1*	
	Мікросхеми		
DD1,DD2 ,DD2.2	Інтегральна мікросхема 74AC109PC	*3*	
DD3-DD6	Інтегральна мікросхема MC74AC163N	*4*	
DD7	Інтегральна мікросхема 74AC04PC	*1*	
	Резистори		
R1	МЛТ-0,5-100 Ом±10%	*1*	
R2-R13	МЛТ-0,5-2 кОм±10%	*12*	
R14	МЛТ-0,5-4,7 кОм±10%	*1*	
R15	МЛТ-0,5-330 кОм±10%	*1*	
	Генератори		
G1	HP8165A	*1*	
G2	AVM-3C	*1*	
	Вимикачі		
SW0	Вимикач живлення	*12*	

ДОДАТОК 2 СХЕМА ЕЛЕКТРИЧНА–ПРИНЦИПОВА



ДОДАТОК 3 СХЕМА СТРУКТУРНА

		Іванов Р.Є.			МРП.171.081.004	Лист
		Писаренко Л.Д.				
Вим.	Арк	№ докум.	Підпис	Дата		



		Іванов Р.Є.			МРП.171.081.004	Лист
		Писаренко Л.Д.				68
Вим.	Арк	№ докум.	Підпис	Дата