**Міністерство освіти і науки України**

**Всеукраїнський конкурс студентських наукових робіт**

Шифр «**Circuit**»

Тема: «**Криптосистема на основі математичної моделі з використанням хаотичних коливань генерованих на базі диференційних рівнянь**»

Спеціальність: **«Інформаційні системи та технології»**

2020

**АНОТАЦІЯ**

наукової роботи під шифром «**Circuit**»

Дана робота присвячена обґрунтуванню застосування криптосистеми, заснованої на математичній моделі генератору хаосу, запропонованого Леоном Чуа у 1983 році, опису принципів реалізації криптоалгоритму та перспективам його застосування.

**Метою роботи** є моделювання генератора хаосу, відомого як «схема Чуа», для його застосування в криптосистемі, надійно функціонує на різних поширених пристроях.

**Завданням роботи** є застосування математичної моделі генератора «схема Чуа» для шифрування даних.

**Об'єктом** дослідження є процес шифрування даних на основі математичної моделі генератора «схема Чуа».

**Предметом** дослідження є математичні моделі, методи, засоби реалізації генераторів хаосу та їх застосування в криптографії.

**Науковою новизною д**аної роботи є розроблений метод використання математичної моделі генератора хаосу «схема Чуа» в якості основного компонента гібридної криптосистеми, де генератор хаосу застосований в якості джерела відкритого та закритого ключів асиметричного алгоритму шифрування і ключ симетричного алгоритму, безпосередньо використовується для шифрування даних.

**Практичне значення** отриманих результатів полягає в створенні програмного забезпечення для шифрування даних на основі «схема Чуа», для якого доведена статистична надійність ключа і зашифрованих даних в бінарному, так і в числовому вигляді.

Результати наукової роботи впроваджено в навчальний процес, що засвідчує акт, та опубліковані в 2 наукових працях, в тому числі 1 наукова стаття у фаховому виданні та 1 публікація у матеріалах конференцій.

Ключові слова: КРИПТОСИСТЕМА, ШИФРУВАННЯ, МАТЕМАТИЧНА МОДЕЛЬ, КРИПТОАЛГОРИТМ, СХЕМА ЧУА.

Зміст

# Вступ

В нинішній час, коли для передачі і зберігання будь-яких даних використовуються поширені і легко доступні технічні засоби, питання захисту інформації від порушення її конфіденційності, цілісності, доступності є однією з найважливіших проблем. На передані дані можуть впливати середовище передачі або зовнішнє (щодо інформаційної системи) середовище, а також різні дії зловмисників, націлені на перехоплення, пошкодження інформації, т. д.

Шифрування переданих даних є одним з методів їх захисту від атак зловмисників і непередбачуваності середовища. Шифрування даних дозволяє підтвердити їх цілісність, забезпечити конфіденційність і доступність інформації для кінцевого одержувача [1].

Для сучасної криптографії характерні відкриті алгоритми шифрування, що використовують обчислювальні засоби. Дані криптоалгоритми припускають наявність ключа певної довжини і сукупність відносно простих перетворень, так званих криптографічних примітивів[2]. Однак, внаслідок природи даних криптосистем, існує велика кількість методів розшифровки закодованої ними інформації.

Однією з основних проблем традиційних криптосистем є те, що, зрештою, послідовність операцій шифрування інформаційного потоку починає повторюватися (довжина ключа є обмеженою), а це призводить до того, що послідовність може виявитися розкрита третьою стороною, а потік – розшифрований. Цього недоліку позбавлені абсолютно стійкі шифри, що задовольняють ряду наступних вимог [3]:

- ключ генерується для кожного блоку шифрованих даних (кожен ключ використовується тільки один раз);

- ключ статистично надійний (ймовірності появи кожного з можливих символів рівні, символи в ключової послідовності незалежні і випадкові);

- довжина ключа дорівнює або більше довжини шифрованих даних;

- вихідний (відкритий) текст має деяку Надмірність (що є критерієм оцінки правильності розшифровки).

Стійкість цих систем не залежить від обчислювальних можливостей криптоаналітика. Проте практичне застосування абсолютно стійких систем обмежено внаслідок складності і вартості їх реалізації, тому в криптографічних системах в основному застосовуються обчислювально стійкі системи, однаково добре функціонують на різних пристроях.

Обчислювально стійка система-це система, з потенційною можливістю розкрити шифр, але тільки при обраних параметрах і ключах шифрування. Стійкість таких систем залежить від обчислювальних можливостей криптоаналітика [4].

**Метою** дослідження є моделювання генератора хаосу, відомого як «схема Чуа», для його застосування в криптосистемі для покращення захисту даних.

Завданням дослідження є застосування математичної моделі генератора «схема Чуа» для шифрування даних.

**Об'єктом** дослідження є процес шифрування даних на основі математичної моделі генератора «схема Чуа».

**Предметом** дослідження є математичні моделі, методи, засоби реалізації генераторів хаосу та їх застосування в криптографії.

**Науковою новизною** даного дослідження є розроблений метод використання математичної моделі генератора хаосу «схема Чуа» в якості основного компонента гібридної криптосистеми, де генератор хаосу застосований в якості джерела відкритого та закритого ключів асиметричного алгоритму шифрування і ключ симетричного алгоритму, безпосередньо використовується для шифрування даних.

**Результати наукової роботи** впроваджено в навчальний процес, що засвідчує акт, та опубліковані в 2 наукових працях, в тому числі 1 наукова стаття у фаховому виданні та 1 публікація у матеріалах конференцій.

# 1 Аналіз предметної області

Предметною областю даної роботи є завдання застосування математичних моделей генераторів хаосу з метою захисту інформації.

Генератор хаосу – динамічна система, що володіє здатністю створювати хаотичний сигнал завдяки своїй крайній чутливість до початкових умов і їх найменших змін.

Динамічний (детермінований) хаос — явище в теорії динамічних систем, при якому поведінка нелінійної системи виглядає випадковим, незважаючи на те, що воно визначається детерміністичними законами.

Ідея використання генераторів хаосу при передачі сигналу не нова. Експерименти по зашифровке і розшифровці сигналів такими методами, які проводилися в 90-х роках 20-го століття, показали перспективність, привабливість і ефективність використання хаотичних генераторів в конфіденційних системах зв'язку [8].

Використання моделей генераторів хаосу, що володіють складним, непередбаченим і сильно залежать від вихідних параметрів поведінкою, як складової програмної криптосистеми може значно підвищити криптостійкість шифру, щодо існуючих на нинішній день алгоритмів.

Однією з головних проблем поширених обчислювально стійких систем, як згадувалося в попередньому розділі, є довжина ключа, пов'язана з методом отримання цього ключа. Кількість операцій і довжина ключа зазвичай жорстко фіксовані криптоалгоритмом. Ключ симетричних криптосистем можна визначити, аналізуючи передаються повідомлення, а ключі криптосистем з відкритим ключем і гібридних криптосистем можна відносно легко визначити при певних параметрах.

Перспективність застосування генераторів хаосу для захисту переданої інформації полягає в трьох особливості хаотичних процесів [9]:

– хаотичний сигнал має неперіодичний, безперервний спектр, що займає досить широку смугу, і його вид можна задавати;

– нерегулярність і непередбачуваність поведінки хаотичного сигналу, а також здатність генератора хаосу створювати абсолютно різні процеси при досить незначній зміні початкових умов, значно утрудняє пророкування процесу на який-небудь тривалий час;

– в силу нерегулярності хаотичних сигналів, їх автокореляційна функція швидко загасає, що також ускладнює передбачення процесу генерації і визначення структури генератора.

У силу всіх цих особливостей, генератори хаосу при належній реалізації алгоритму, можуть бути основою таких генераторів випадкових послідовностей чисел, ступінь непередбачуваності яких буде досить високою, близькою до істинно випадкових послідовностей, де результат практично не передбачуваний незалежно від наявності інформації про алгоритм і параметрах.

## 1.1 Аналіз методів рішення

Існує велика кількість різних математичних моделей фізичних процесів, що володіють хаотичним поведінкою. Однак найбільш придатними для використання в криптосистемах є моделі таких дисипативних (характеризуються спонтанним виникненням складної структури, не зберігає обсяг у фазовому просторі систем з хаотичною динамікою, де спектр одержуваних значень генератора можна задавати і змінювати режими роботи моделі в досить широкому діапазоні [10]. До таких генераторів хаосу відноситься, наприклад, схема Чуа.

Оскільки схема Чуа є досить добре дослідженої моделлю динамічного хаосу, володіє складною поведінкою при загальній простоті реалізації і здатна працювати в широкому діапазоні значень, вона була обрана в якості об'єкта моделювання з метою використання її для захисту інформації.

Схема Чуа дозволяє при зміні параметрів схеми відтворювати поведінку таких різних хаотичних множин, атракторів, як аттрактор Лоренца, аттрактор Ресслера та інших.

Окремі математичні моделі таких хаотичних множин зазвичай сильно обмежені по кількості і різноманітності їх режимів роботи, так як їх параметри і структура досить обмежені, будучи результатами різних експериментів (наприклад, аттрактор Лоренца було отримано у результаті дослідження гідродинаміки [11]). Внаслідок цього вони не можуть бути оптимальним рішенням для хаотичного шифрування даних, так як такі обмеження параметрів і структури роблять їх більш уразливими з точки зору можливості визначення цих параметрів і структури при аналізі зашифрованого сигналу або перехоплення ключів синхронізації генераторів. Іншими словами, дані моделі можуть виявитися занадто передбачуваними в реальних умовах.

У цьому сенсі генератори хаосу зразок схеми Чуа значно менш передбачувані, так як здатні, як вже було сказано раніше, функціонувати в різних режимах і відтворювати поведінку різних хаотичних моделей.

## 1.2 Огляд існуючих рішень та програм

Серед рішень задачі про захист інформації за допомогою генераторів Хаосу існують апаратні та програмні рішення. Однак, слід зазначити, що ці деякі апаратні і програмні хаотичні криптосистеми, заснованих на різних моделях генераторів Хаосу, широкого поширення не отримали і з метою шифрування для зберігання або передачі цифрових даних ці рішення в нинішній час не застосовуються.

Пошук інформації про такі програмні рішення в цілому зводиться до кількох закордонних проектів, у тому числі наукових статей [12], і патентів [13]. У нашій країні програмних розробок і практичних реалізацій щодо

використання генераторів хаосу у сфері захисту цифрової інформації знайдено не було, проте були теоретичні дослідження, на базі яких проводилося моделювання і висувалися методи синхронізації генераторів [14][15].

Розглянуті проекти криптосистем на генераторах Хаосу використовують для синхронізації генераторів різні значення, що передаються по секретних або відкритих каналах. Використання секретних каналів на практиці передбачає додаткове шифрування цих значень, що ускладнює реалізацію криптосистеми. Дану проблему можна обійти, зменшивши кількість переданих значень і встановивши залишилися значення в якості початкових констант. В такому випадку, генератор Хаосу використовується як незворотна функція за аналогією з криптосистемами з відкритим ключем, проте в такому випадку існує можливість отримати дані синхронізації, перехопивши відкритий ключ, і, відповідно, виконати розшифровку, підміну або спотворення переданих даних.

У випадку традиційних криптосистем з відкритим ключем вказана проблема вирішується за допомогою сертифікатів та певного набору відкритих ключів, але такий підхід не дозволить реалізувати весь потенціал криптосистеми на генераторі хаосу, де кількість відкритих ключів або наборів початкового стану генератора не повинна бути обмежена. Обмеження кількості можливих початкових значень може значно позначитися на унікальності генерується послідовності випадкових чисел.

## 1.3 Математична модель

У проекті криптосистеми використовується програмна реалізація математичної моделі схеми Чуа, одного з видів генераторів хаосу. Схема Чуа (або ланцюг Чуа) — це проста електронна схема, що демонструє ряд біфуркаційних явищ і атракторів. Ланцюг складається з двох конденсаторів,

котушки індуктивності, лінійного резистора і нелінійного резистора (зазвичай званого діодом Чуа)[17]. Електрична принципова схема ланцюга Чуа в класичному варіанті представлена на рисунку 1.

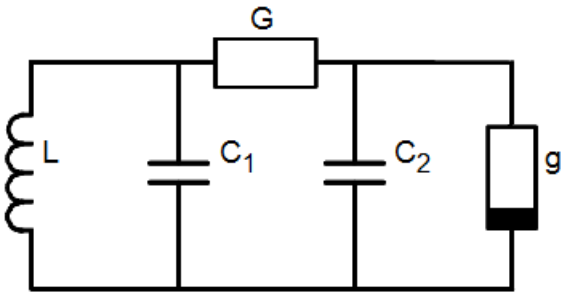


Рисунок 1 – Схема Чуа; L – котушка індуктивності, G – резистор, C1 і C2 – конденсатори, g – нелінійний опір (діод Чуа); параметри елементів: L=1/7Гн; G=0.7 См; C1=1/9Ф; C2=1Ф

Складність структури хаотичних сигналів і їх нерегулярність, а також можливість створювати одним і тим же хаотичним генератором абсолютно різні процеси при досить незначній зміні початкових умов, надає можливість створювати моделі генераторів хаосу для їх застосування в криптографії, чому присвячена дана робота.

Завдяки тому факту, що схема Чуа є одним з найпростіших генераторів хаосу (її математична модель описується трьома диференційними рівняннями, але при цьому володіє складною поведінкою, властивим генераторів хаосу), вона була обрана для системи шифрування інформації як найбільш ефективна з точки зору реалізації. Схема Чуа описується наступною системою рівнянь [18]:

$$\begin{cases} C_1 \dfrac{dv_{C_1}}{dt} = G\left(v_{C_2} - v_{C_1}\right) - g\left(v_{C_1}\right) \\ C_2 \dfrac{dv_{C_2}}{dt} = G\left(v_{C_1} - v_{C_2}\right) - i_L \\ L \dfrac{di_L}{dt} = -v_{C_2} \end{cases}, \quad (1.1.1)$$

де $g(v_{C_1})$ – кусково-лінійна функція, визначена як:

$$g(v_{C_1}) = G_b v_{C_1} + \frac{1}{2}(G_a - G_b)\left(|v_{C_1} + E| - |v_{C_1} - E|\right) \qquad (1.1.2)$$

.

Нелінійна функція 1.1.2 представлена графічно на рисунку 2: крутизна внутрішнього і зовнішнього ділянок Ga і Gb, відповідно; при цьому точки E відповідають зламів на графіку [18].

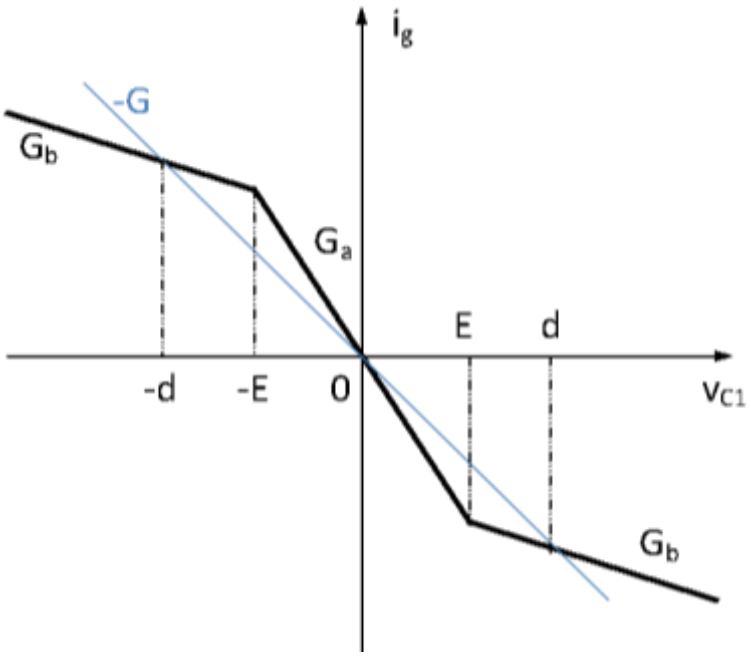


Рисунок 2 – Графік вольт-амперної характеристики діода Чуа

На графіку також показана навантажувальна пряма, від перетину з якою утворюються три точки рівноваги в, 0 і-в.

Після заміни коефіцієнтів в системі рівнянь 2.1.1 на безрозмірні, система прийме наступний вигляд [18]:

$$\begin{cases} \frac{dx}{dt} = a(y - x - h(x)) \\ \frac{dy}{dt} = x - y + z \\ \frac{dz}{dt} = -\beta y \end{cases}, \tag{1.1.3}$$

де $h(x)$ визначено як:

$$h(x) = m_1 x + \frac{1}{2}(m_0 - m_1)(|x+1| - |x-1|). \tag{1.1.4}$$

В рамках даної роботи була використана модифікована математична модель, де доданий коефіцієнт c, що дозволяє більш гнучко управляти зміною режимів коливань системи. Модифікована система рівнянь має наступний вигляд:

$$\begin{cases} \frac{dx}{dt} = a(y - x - h(x)) \\ \frac{dy}{dt} = c(x - y + z) \\ \frac{dz}{dt} = -\beta y \end{cases}. \tag{1.1.5}$$

Функція $h(x)$ в системі 1.1.5 відповідає своєму визначенню з формули 1.1.4.

Чисельне рішення цих рівнянь показує, що при певних співвідношеннях між компонентами ланцюга, зміна значень змінних в часі набуває хаотичний характер. Однак хаотичні режими коливань виявляються в досить вузькій області параметрів. Основні режими коливань умовно показані на рисунку 3 [18].

У випадку, коли параметри системи знаходяться в області позначеної цифрою 2 в околі точки рівноваги d або d існує стійкий граничний цикл. По мірі наближення до границі з хаотичним режимом, система зазнає цикл подвоєнь періоду аж до утворення хаотичного атрактора Ресслера [19].

При попаданні параметрів в область позначену цифрою 6 утворюється дивний атрактор, званий «подвійний завиток», показаний на рисунку 4 (випадок для моделі з безрозмірними коефіцієнтами) [18].

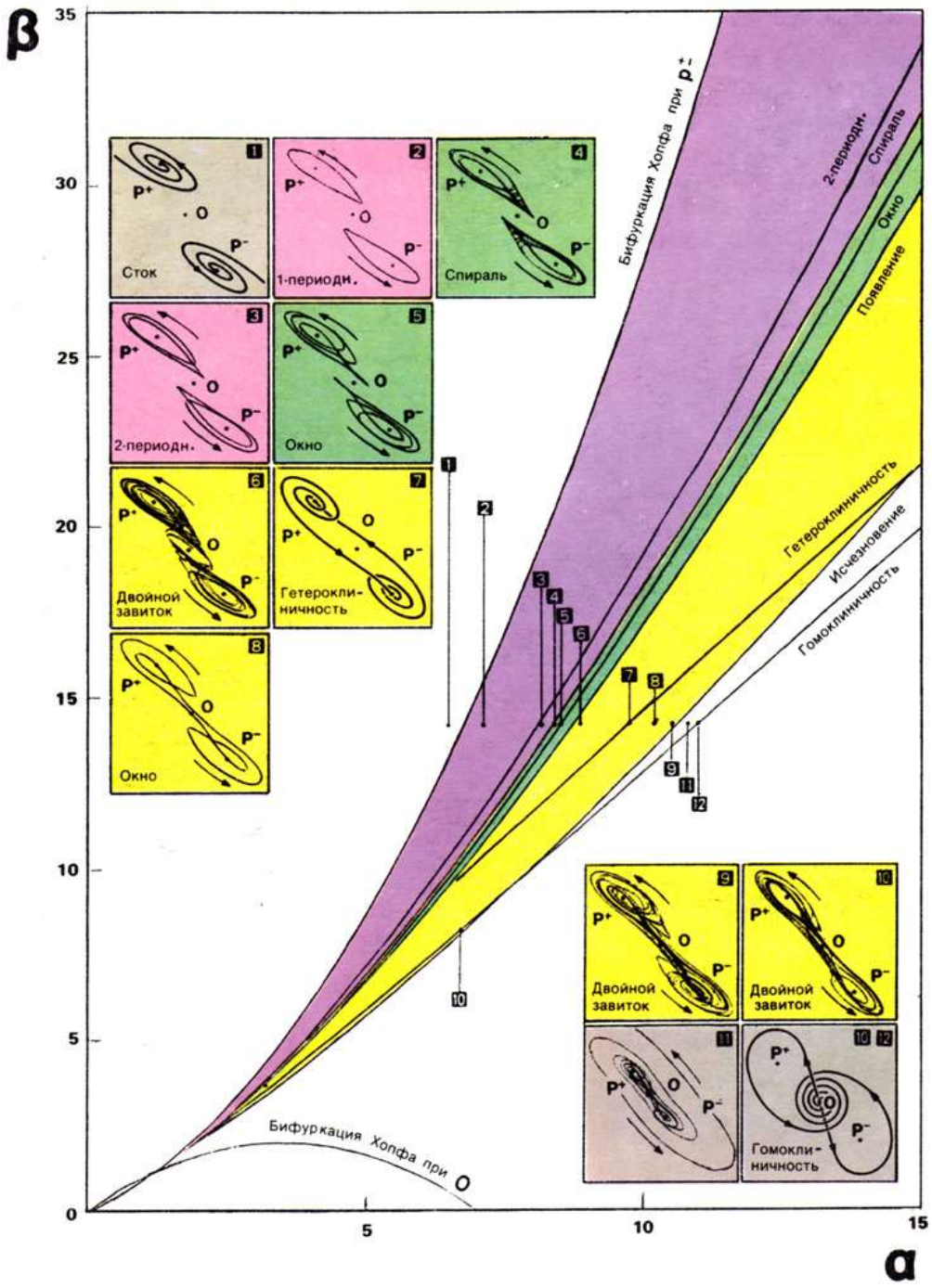


Рисунок 3 – Біфуркаційна діаграма режимів при m0=-8/7, m1=-5/7, c=1

Траєкторія такого аттрактора неперіодична і режим функціонування нестійкий, в результаті чого навіть малі відхилення параметрів викликають

значні зміни. Результатом такої поведінки є не періодичність в часі будь-який з координат системи, суцільний спектр і спадна в часі автокореляційна функція [19].

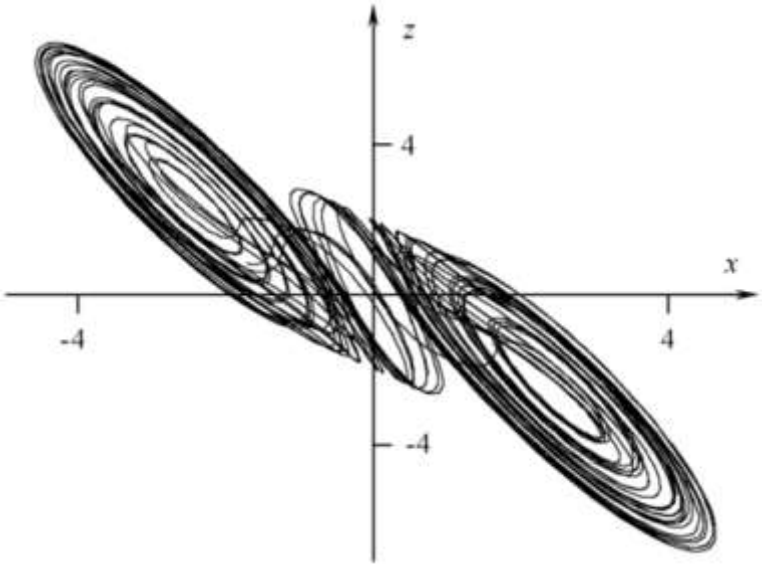


Рисунок 4 – Аттрактор виду «подвійний завиток»

Це викликає хаотичність динаміки дивних атракторів, а саме вказує на те, що прогнозування траєкторії, що потрапила в атрактор, утруднене, оскільки мала неточність у початкових даних через деякий час може призвести до сильного розбіжності прогнозу з реальною траєкторією.

На рисунку 5 показана одна з можливих часових залежностей зміни значення x для моделі з безрозмірними коефіцієнтами [18].

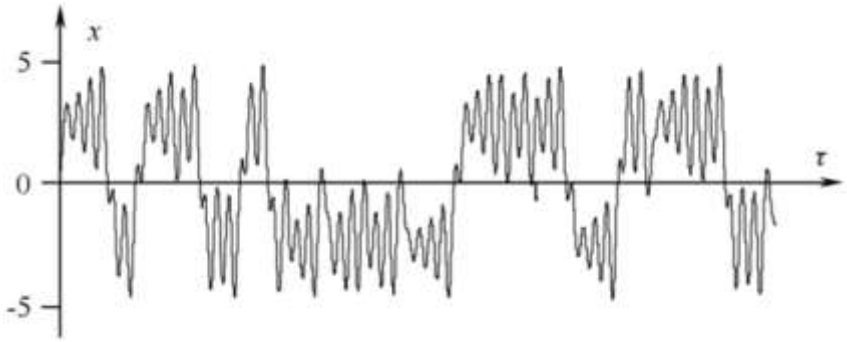


Рисунок 5 - Тимчасова залежність зміни x

Тимчасова залежність зміни x є одним з параметрів, який може бути використаний в криптосистемі для зашифровки переданої інформації, наприклад, шляхом накладення хаотичного сигналу на інформаційний, гамування.

У загальному випадку шифрування може здійснюється за допомогою декількох алгоритмів [11]:

– хаотична маскування – інформаційний сигнал підсумовується з хаотичним сигналом;

– перемикання режимів – логічний нуль кодується одним типом хаотичного сигналу (наприклад, отриманим значенням x), логічна одиниця – іншим (наприклад, значенням y);

– нелінійне підмішування – інформаційний сигнал бере участь у формуванні самого хаотичного сигналу.

В рамках даної роботи застосовано алгоритм хаотичного маскування з використанням змінних у часі вихідних значень x генератора хаосу, причому для досягнення мірою непередбачуваності, близькою до істинно випадкових послідовностей, виконаний зріз значень параметрів, умовно показаний на рисунку 6.

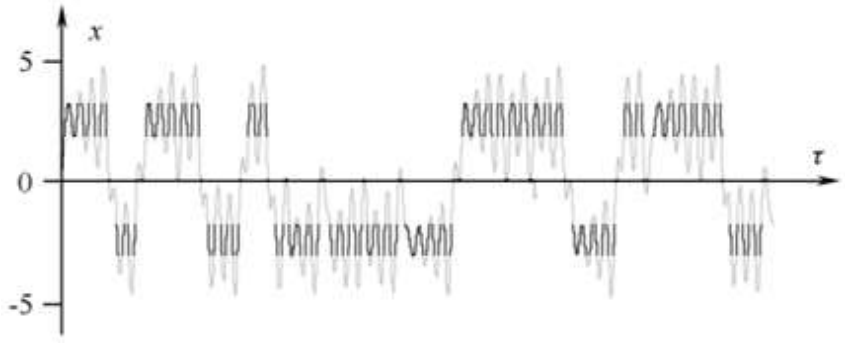


Рисунок 6 – Зріз значень параметра x

Як видно з рисунка 6, всі точки графіка, де результат функції зустрічається набагато рідше (сірі лінії графіка), були відкинуті, а решту

значення x (чорні області) мають рівну статистичну частоту входження, де генерується ключ. З урахуванням неперіодичности і непередбачуваності, генератор хаосу в даному випадку виступає в якості генератора істинно випадкових чисел на двох проміжках значення x.

Параметри початку генерації ключа криптосистемой, необхідні для синхронізації генераторів хаосу, вибираються випадковим чином з метою забезпечення його унікальності для кожного нового сеансу передачі даних.

# 2 Проектна модель

При проектуванні застосовувалися методи об'єктно-орієнтованого аналізу і проектування, тому що розробляється система спочатку передбачалася бути об'єктно-орієнтованою.

Призначення об'єктно-орієнтованого аналізу і проектування полягає в розгляді предметної області і логічного вирішення завдання з точки зору об'єктів (понять і сутностей). В процесі об'єктно-орієнтованого аналізу основна увага приділяється визначенню та опису об'єктів (або понять) в термінах предметної області.

В процесі об'єктно-орієнтованого проектування визначаються логічні програмні об'єкти, які будуть реалізовані засобами об'єктно-орієнтованої мови програмування. Ці програмні об'єкти включають атрибути та методи. У процесі конструювання або об'єктно-орієнтованого програмування забезпечується реалізація розроблених компонентів і класів [20].

## 2.1 Діаграма варіантів використання

Діаграма варіантів використання, представлена на малюнку 2.7, відображає основні можливості розроблюваної системи.

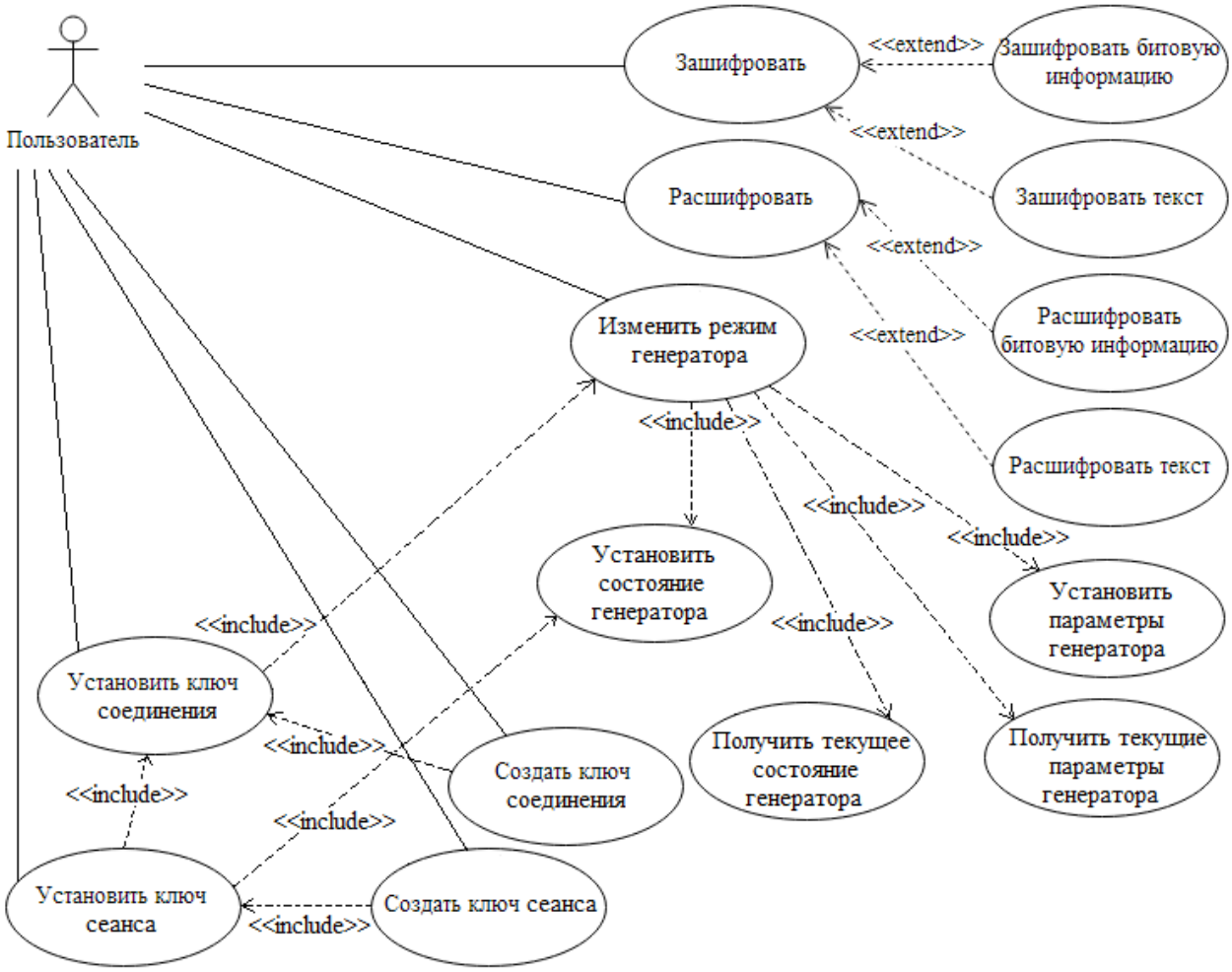


Рисунок 7 – Діаграма варіантів використання

З діаграми варіантів використання видно, що синхронізація криптосистем відправника і одержувача здійснюється за допомогою як сеансового ключа, так і ключа з'єднання. Ключ з'єднання використовується для однозначної взаємної ідентифікації пари пов'язаних між собою примірників криптосистеми. Таким чином, перехоплення сеансового ключа не призведе до розшифровки переданих даних.

На діаграмі варіанти використання «Зашифрувати» і «Розшифрувати» показано як різні дії, проте в даному випадку реалізації алгоритму шифрування для зашифровки і розшифровки буде використовуватися одна і та ж функція. Це пов'язано з тим фактом, що в шифрующем алгоритмі передбачається використання хаотичного маскування, заснованої на операції

виключного "АБО" (XOR) [21], де для шифрування і розшифрування можливе використання однієї і тієї ж функції.

XOR дозволяє як додати ключ до даних, так і видалити його без змін в коді (на відміну від, наприклад, операції додавання "I", що вимагає подальшого віднімання для відновлення вихідних даних). Дії з текстової інформації і бітової є різними варіантами використання, тому що припускають різну реалізацію. Таким чином, основні можливості системи були розглянуті.

## 2.2 Опис варіанту використання

Варіант використання — можливість модельованої системи (частина її функціональності), завдяки якій користувач може отримати конкретний, вимірюваний і потрібний йому результат [20]. Варіант використання відповідає окремій функції системи, визначає один з варіантів її використання і описує типовий спосіб взаємодії користувача з системою.

Розглянутим варіантом використання є варіант використання «Розшифрувати текст». Метою варіанту використання є розшифровка тексту. Дійовою особою є користувач програми. Попередніми умовами є наявність ключа з'єднання і сеансового ключа. Активатором є команда користувача на перевірку наявності зашифрованого тексту.

Основний потік подій:

1) Криптосистема встановлює параметри і стан генератора;

2) Криптосистема отримує зашифрований текст;

3) Текст розшифровується;

4) Перевіряється контрольна сума;

5) Розшифрований текст виводиться.

Альтернативні сценарії:

1) Не встановлено або некоректний ключ з'єднання: повернення коду помилки про відсутність ключа з'єднання;

2) Не встановлено або некоректний ключ сеансу: повернення коду помилки про відсутність сеансового ключа;

3) Помилка при отриманні даних: повернення коду помилки про помилку;

4) помилка під час перевірки контрольної суми тексту: повернення коду помилки про некоректний блок тексту.

## 2.3 Діаграма класів

Діаграма класів призначена для демонстрації класів, їх атрибутів, методів і взаємозв'язків між ними [20].

Діаграма класів, що забезпечують функціонування моделі генератора хаосу і криптосистеми, наведена на рисунку 8.

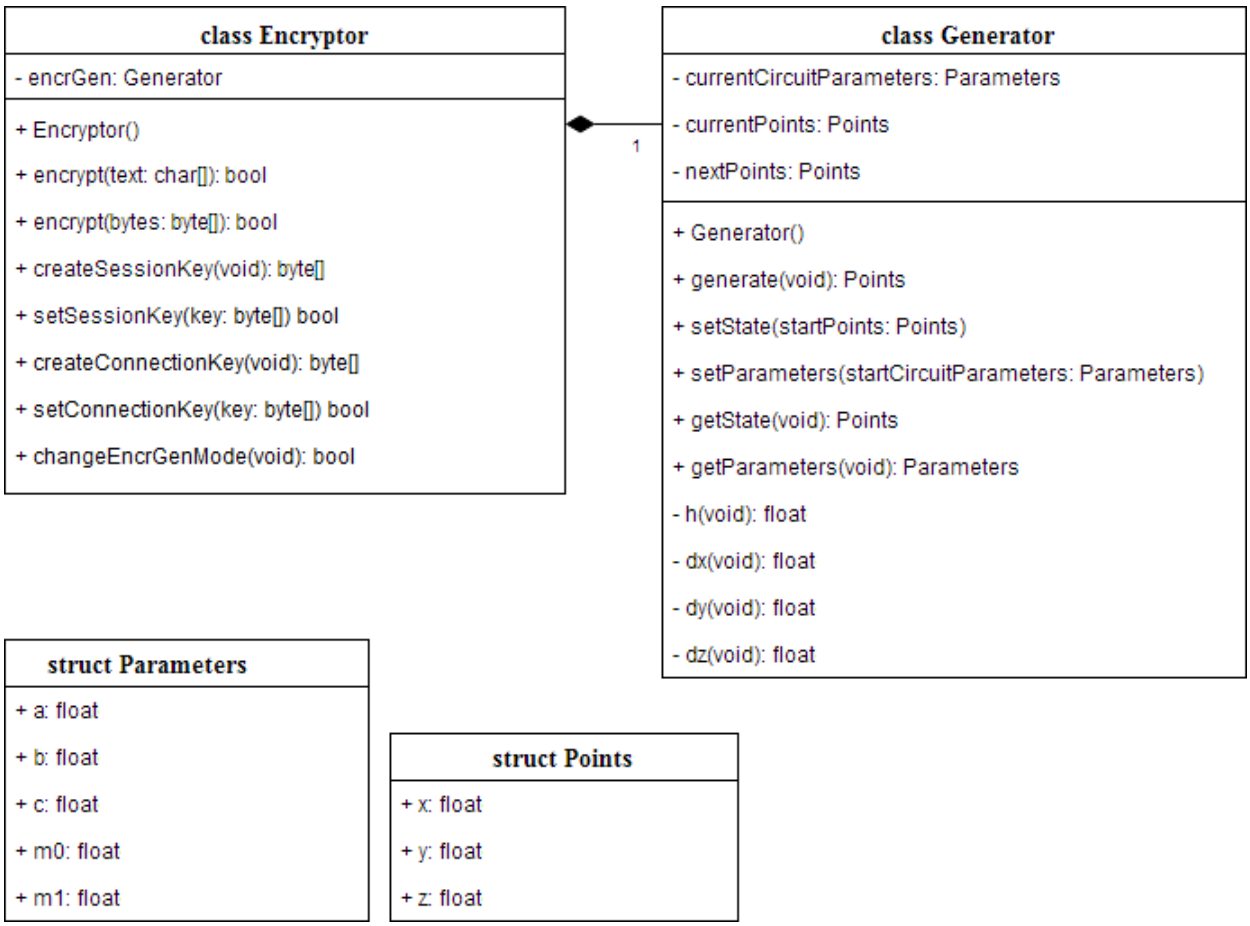


Рисунок 8 – Діаграма класів

## 2.4 Діаграма послідовності

Діаграма послідовності призначена для відображення набору об'єктів на єдиній часовій осі їх життєвого циклу (створення-діяльність-знищення) і взаємодії (відправка запитів і отримання відповідей) [20].

Діаграма послідовності для розглянутого варіанту використання «Розшифрувати текст» представлена на рисунку 9.

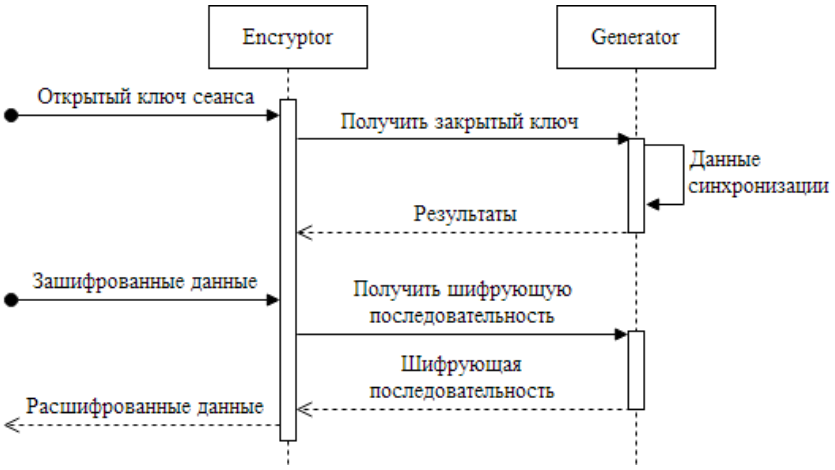


Рисунок 9 – Діаграма послідовності для варіанту використання «Розшифрувати текст»

З діаграми послідовності видно, які функції виконуються об'єктом класу Encryptor, які виконуються класом Generator і в якому порядку.

## 2.5 Діаграма діяльності

Діаграма діяльності — UML-діаграма, на якій показано розкладання деякої діяльності на її складові частини. Під діяльністю розуміється специфікація виконуваного поведінки у вигляді координованого послідовного і паралельного виконання підлеглих елементів — вкладених видів діяльності та окремих дій, з'єднаних між собою потоками, які йдуть від виходів одного вузла до входів іншого [20]. Діаграма діяльності для розглянутого варіанту використання «Розшифрувати текст» представлена на рисунку 10.

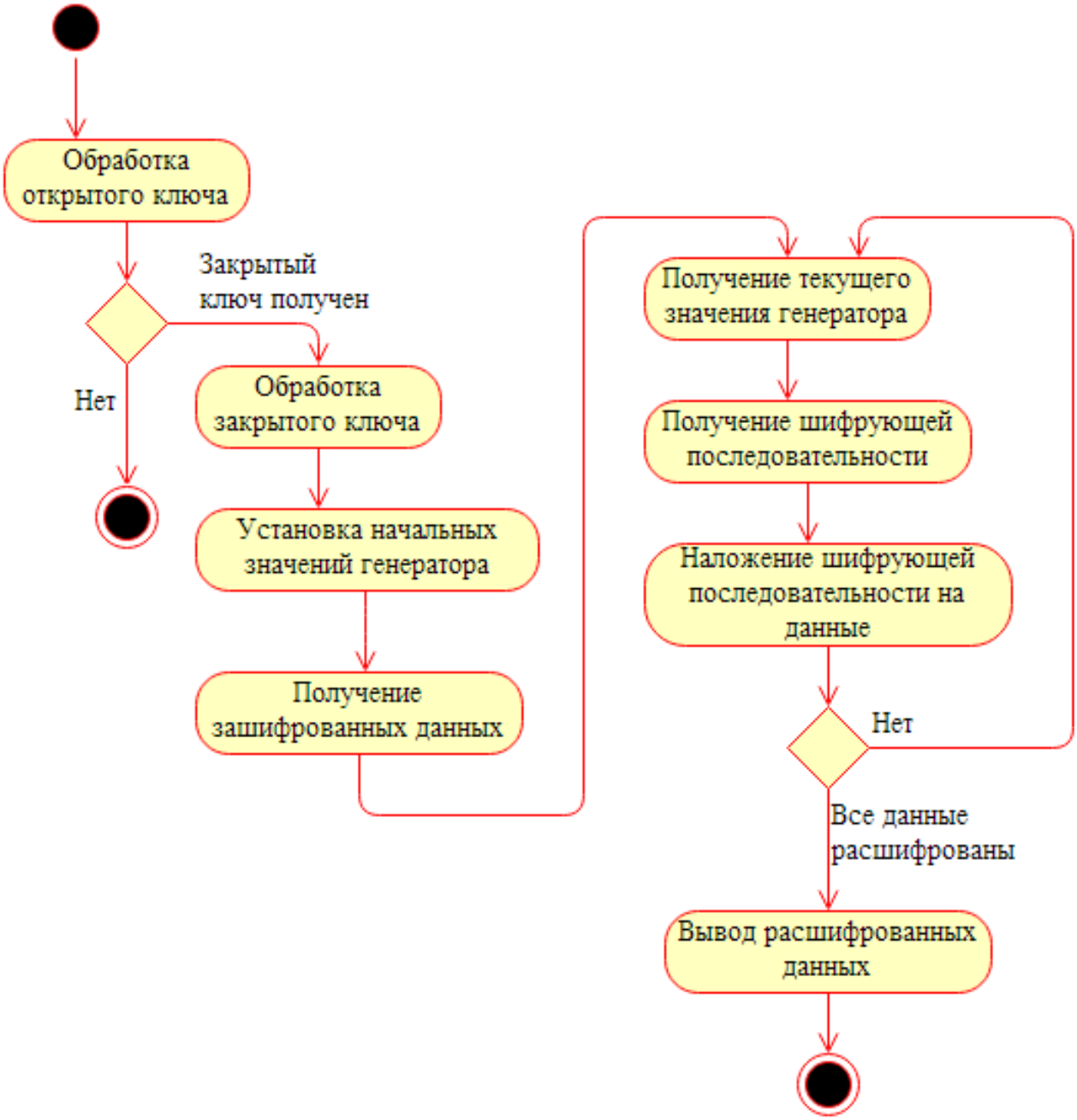


Рисунок 10 – Діаграма діяльності для варіанту використання «Розшифрувати текст»

З діаграми діяльності видно, з яких частин складається і як взаємопов'язані частини діяльності для варіанту «Розшифрувати дані». Як показано на діаграмі, шифрувальник послідовність генерується кожен раз заново для окремого блоку даних. Ця особливість алгоритму пов'язана з тим фактом, що шифрує дані ключ генерується безпосередньо в процесі шифрування без виділення під нього такого нового блоку пам'яті, який би містив увесь ключ, рівний довжині оброблюваних даних. Такий підхід дозволяє зменшити

кількість використовуваної системою оперативної пам'яті ПК або іншого пристрою, так як довжина поточного блоку даних передбачається значно меншою, ніж вся довжина даних, і рівній довжині згенерованого для блоку ключа.

# 3 Програмне забезпечення експерименту

Для реалізації математичної моделі генератора хаосу і криптосистеми на її базі, був обраний мова програмування C++. Дана мова програмування є об'єктно-орієнтованим, поєднує властивості як високорівневих, так і низькорівневих мов, володіє широкими можливостями, є велика кількість довідкової та навчальної літератури з мови. Так як розроблювана система на стадії проектування передбачалася бути об'єктно-орієнтованої, реалізація також виконана за допомогою методів об'єктно-орієнтованого програмування.

В якості середовища програмування була обрана RAD Studio. Середовище підтримує роботу з такими мовами програмування, як C, C++, C#, Delphi. Крім того, середовище програмування добре документована, існує велика кількість розробників, що використовують цю середу, і спільнот, присвячених їй.

На рисунку 11 (Додаток 1) наведено фрагмент програмного коду програми, що використовує розроблену в рамках дипломної роботи криптосистему на базі генератора хаосу «схема Чуа».

На наведеному рисунку фрагмент коду відноситься до блоку рішення системи диференціальних рівнянь моделі генератора хаосу (для вирішення використовується метод Рунге-Кутта [22]) і зрізу одержуваних значень, методи встановлення і отримання поточних параметрів і значень генератора, фрагмент методу зашифровки текстових даних.

Розроблений додаток являє собою пару незалежних один від одного процеси. Інтерфейс цих процесів, EncryptorTestC і EncryptorTestS, є практично ідентичним і наочно демонструє результати роботи криптосистеми. Вікно провідного процесу EncryptorTestC представлено на рисунку 12 (Додаток 1).

Процес EncryptorTestC є провідним процесом програми та містить ведучий генератор хаосу, з яким синхронізується ведений генератор хаосу в процесі EncryptorTestS.

При натисканні кнопки «Відправити» у формі, текст, введений у відповідне поле на формі EncryptorTestC, відображається у двох видах в полях, що належать до надісланих повідомлень, і далі у зашифрованому вигляді передається процесу EncryptorTestS для розшифровки, відображення, зашифровки і передачі його у вигляді exo-повідомлення.

Сам механізм обробки введених в додаток даних працює наступним чином. Дані, які потрібно зашифрувати і передати, зберігаються в тимчасовому масиві, на базі розміру якого визначається, яка довжина ключа потрібно для зашифровки тексту. Далі, на блоки оброблюваного тексту за допомогою операції виключає «АБО» (XOR) накладається генерується ключ, що відповідає розміру блоку довжини. Ключ генерується на базі результатів роботи генератора хаосу «схема Чуа», а саме зміни значень змінної x у часі. Згенеровані числа за допомогою операцій бітового зсуву перетворюються в шифруючу послідовність, яка і накладається на поточний блок тексту.

Шифрувальник функція криптосистеми працює безпосередньо з переданим їй тимчасовим масивом тексту. Зашифрований текст в бітовому вигляді записується в файл, який відкриває для обробки процес EncryptorTestS, вікно якого зображено на рисунку 13 (Додаток 2). Отримавши зашифровані дані, EncryptorTestS виконує зворотне перетворення зашифрованого тексту аналогічним зашифровке способом, так як внаслідок особливостей роботи операції XOR, підмішування і видалення ключа з зашифрованих даних може здійснювати одна і та ж функція без зміни

програмного коду. Exo-повідомлення процесу EncryptorTestS, адресоване EncryptorTestC, зашифровується, відправляється і розшифровується аналогічним способом.

Для передачі зашифрованих даних між собою програми використовують запис і читання бінарного файлу, що було зроблено для можливості аналізу результатів роботи криптосистеми. Передача ключів синхронізації здійснена аналогічним способом.

В результаті, розроблене додаток являє собою зручну модель для проведення експериментальних досліджень конкретної реалізації математичної моделі генератора хаосу «схема Чуа» та криптосистеми, заснованої на даній моделі.

## 3.1 Опис експерименту

Суть проведеного експерименту полягає в аналізі основи генерованого ключа і її статистичної перевірки на відповідність необхідним параметрам розподілу. В даному експерименті, генератор Хаосу розглядається як генератор випадкових послідовностей і чисел, які використовуються криптосистемою при створенні ключа. Статистична перевірка послідовностей і розподілу чисел у вибірці показує, наскільки змодельований генератор хаосу відповідає вимогам щодо генерації істинно випадкових чисел і унікальних неповторюваних послідовностей [23].

На малюнку 14 показаний фрагмент гістограми частоти появи чисел, отриманих за допомогою генератора хаосу, у вибірку. У випадку даного експерименту, розмір вибірки склав 1048576 чисел (що з урахуванням алгоритму створення шифрувальної послідовності відповідає ключу довжиною приблизно 800 КБ).

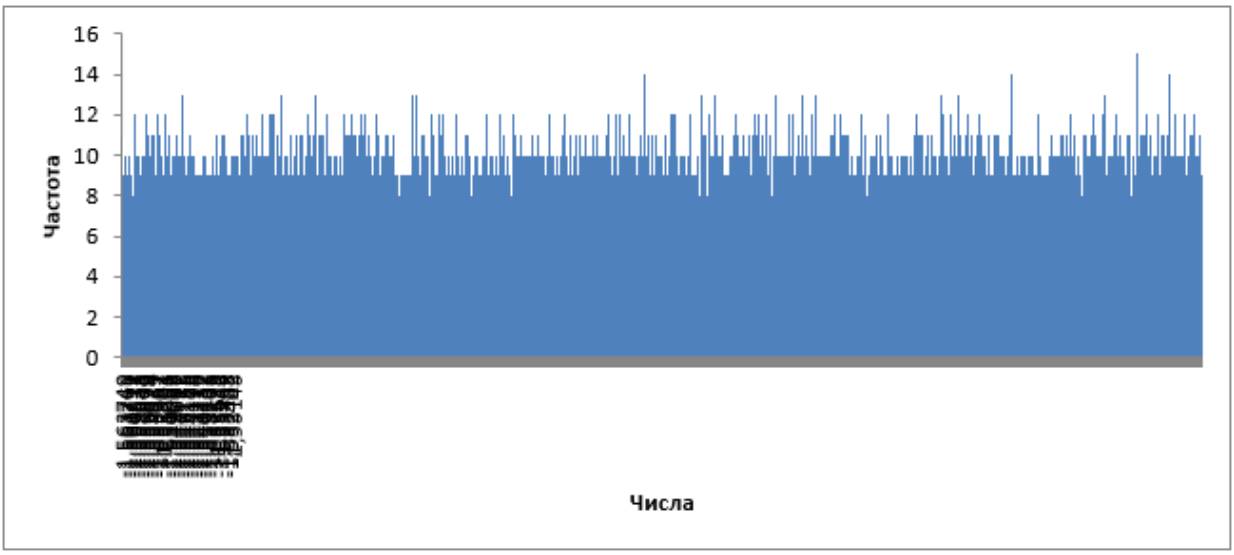


Рисунок 14 – Частота появи у вибірку 1048576 чисел, отриманих за допомогою генератора хаосу

Як видно з малюнка 14, частота появи різних чисел діапазону від -1,6 до -1,3 і від 1,3 до 1,6 у вибірку склала від 1 до 14 входжень. Число входжень відносно невелика і розподілено досить рівномірно, що є хорошим результатом з урахуванням розмірів вибірки і обмежень діапазону і точності. В неперервних випадкових вибірках розміром близько 500 чисел, не знайдено більше одного входження будь-якого числа (рисунок 15).

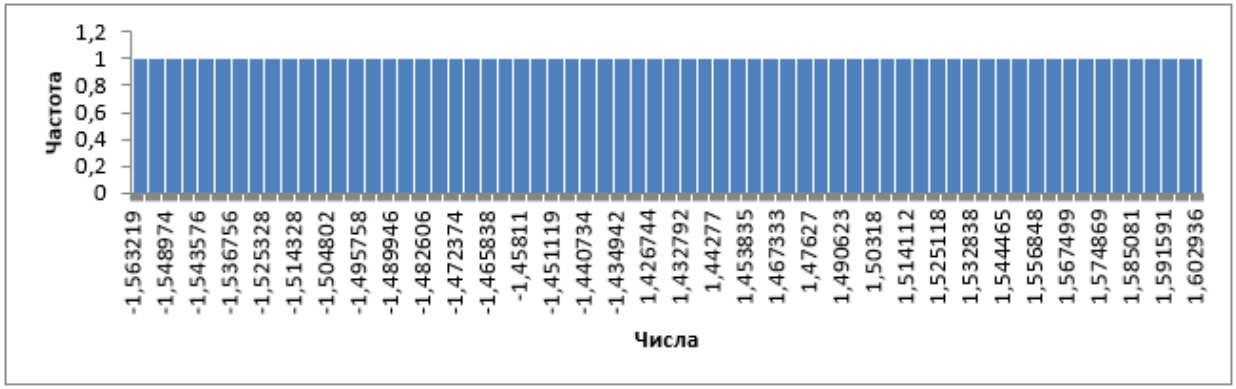


Рисунок 15 – Частота появи у вибірку 499 чисел, отриманих за допомогою генератора хаосу

На малюнку 16 показана гістограма частоти входжень чисел в безперервну випадкову вибірку довжиною в 1143 числа.

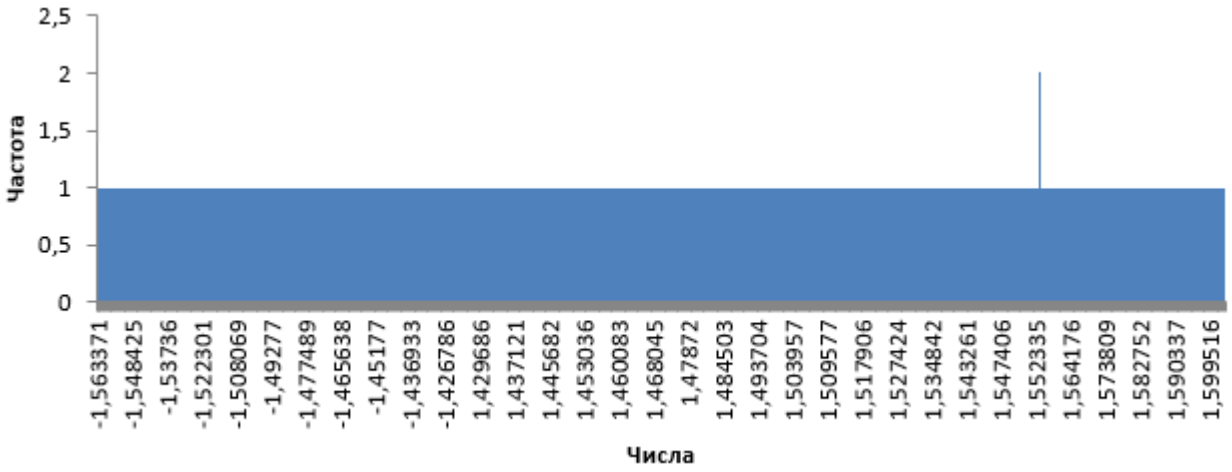


Рисунок 16 – Частота появи в безперервну випадкову вибірку 1143 чисел

З рисунку 16 видно, що частота входження збільшилася тільки для одного числа на даному проміжку. Також було проведено ще кілька випадкових вибірок на базі загальної вибірки довжиною в 1048576 чисел. Так результат випадкової вибірки з 651 випадково обраного числа показаний на малюнку 17.

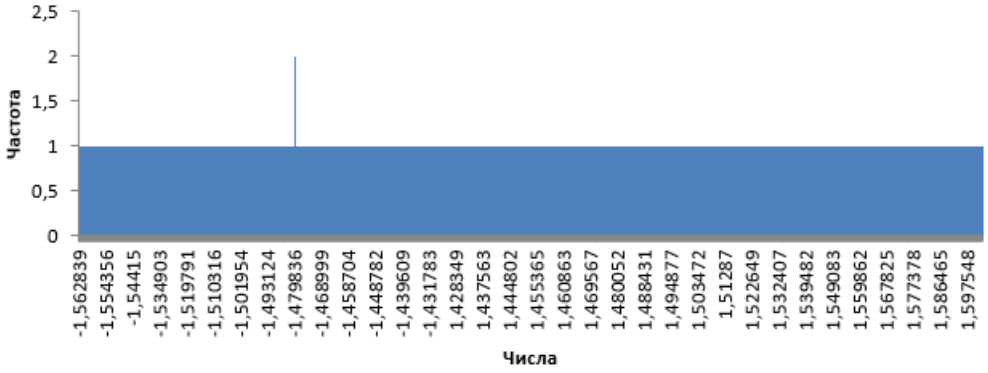


Рисунок 17 – Частота появи у вибірку 651 випадково обраного числа

З рисунку 17 видно, що частота входження у вибірку збільшилася тільки для одного числа.

Результат випадкової вибірки з 2041 випадково вибраних чисел показаний на малюнку 18.

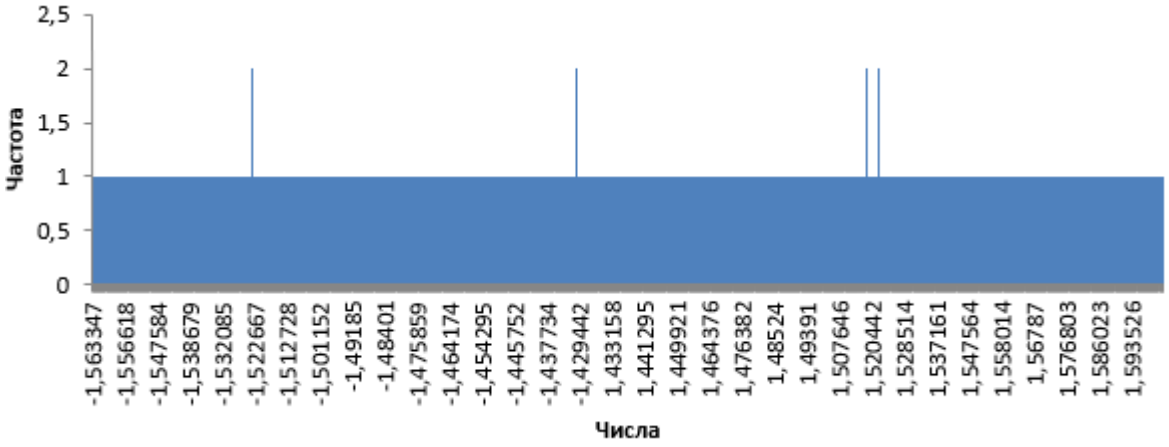


Рисунок 18 – Частота появи у вибірку 2041 випадково обраного числа

З рисунку 18 видно, що при випадковому виборі чисел по всьому протягу початкової вибірки, частота входження збільшилася тільки для декількох окремих чисел, і не перевищує двох. З усього цього випливає, що рівномірність розподілу не залежить безпосередньо від кількості чисел, ні від їх порядку у вибірці.

Також, у всій тестованої вибірці (1048576 чисел) не було знайдено повторень жодної близької пари чисел, що підтверджує те, що послідовність неперіодична і непередбачувана.

## 3.2 Статистична перевірка ключа і зашифрованих даних

Для виконання статистичної перевірки згенерованого ключа і зашифрованих даних використаний пакет статистичних тестів Diehard [23][24]. Серед тестів даного пакета є тести як числових, так і бітових послідовностей, що важливо при тестуванні ключа криптосистеми і

зашифрованих нею даних. Для тестування була взята вибірка бітових даних розміром 11 МБ.

У таблиці 1 (Додаток 3) наведено результати статистичного тестування ключа криптосистеми.

Згідно з таблицею, майже всі тести були пройдені успішно. Тести «Потік бітів» і «Тест стиснення» пройдені не були, однак, варто відзначити, що результати близькі до значень критерію проходження тесту – потрапляння значення в інтервал [0,025; 0.975].

У таблиці 2 (Додаток 4) наведено результати статистичного тестування зашифрованих даних.

Згідно таблиці, майже всі тести для зашифрованих даних також були пройдені успішно. Успішне проходження майже всіх тестів, як ключем, так і зашифрованими даними, вказує на те, що обидва набору, ключ і дані, є статистично випадковими. На базі цього можна стверджувати, що з боку ключ і дані виглядають абсолютно випадковим набором біт.

## 3.3 Тестування стабільності системи

Даний тест призначений для оцінки стабільності роботи розробленої криптосистеми і генератора хаосу в її основі. Суть тесту полягає в тому, що потрібно зашифрувати і розшифрувати деяку кількість тестових даних різного об'єму при різних режимах роботи генератора. В якості тестових даних для наочності були обрані фрагменти тексту.

На рисунку 19 представлений графік стабільності розшифровки безперервного блоку даних при режимі роботи генератора за замовчуванням і при перемиканні режимів, де вісь ординат являє собою відсоток успішно розшифрованих символів, а вісь абсцис – кількість символів + вибірка.

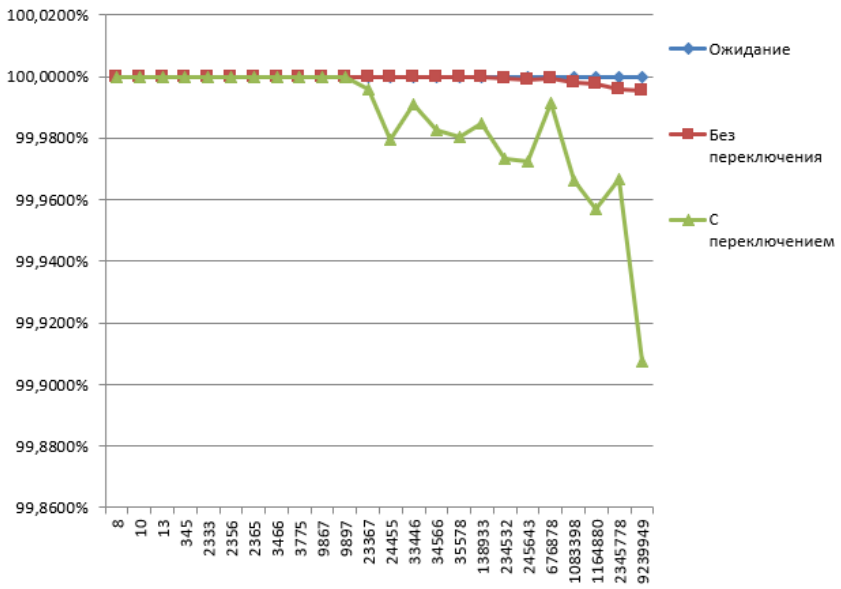


Рисунок 19 – Стабільність розшифровки даних

Згідно з графіком, кількість вірно розшифрованих символів при роботі генератора в штатному режимі без перемикання практично відповідає кількості символів, поданих системі. Незначна розбіжність починає спостерігатися при роботі з текстами, обсяг яких становить понад 200 тисяч символів.

При роботі генератора з перемиканням режимів похибка виявлена під час розшифровки текстів, об'ємом більше 20 тисяч символів. Похибка при роботі з великими об'ємами даних в цьому випадку далека від лінійності, що пов'язано з тим фактом, що при різних режимах роботи генератора і переключення між ними можуть виникати помилки.

В рамках експериментальних досліджень успішно реалізовано додаток, що використовує розроблену криптосистему на базі генератора хаосу «схема Чуа». Розроблений додаток являє собою зручну модель для проведення експериментальних досліджень конкретної реалізації криптосистеми.

Проведено ряд статичних тестів і виконано тестування стабільності роботи реалізованої математичної моделі генератора хаосу «схема Чуа» і самої розробленої криптосистеми. Таким чином, експериментальні дослідження завершені.

## Висновки

В результаті виконання наукового проекту був виконаний аналіз предметної області і конкретного завдання, існуючих рішень і проектів, розглянуто традиційні криптосистеми і проекти криптосистем на генераторах хаосу.

На базі літературних джерел, була розглянута математична модель генератора хаосу "схема Чуа. У результаті була розроблена математична модель для конкретного програмного рішення задачі про захист даних.

За допомогою методів об'єктно-орієнтованого аналізу і проектування, розроблена проектна модель криптосистеми, що описує класи та їх взаємодію між собою у часі, наведено поведінка криптосистеми для одного з конкретних випадків.

В рамках експериментальних досліджень успішно реалізовано додаток, що використовує розроблену криптосистему на базі генератора хаосу «схема Чуа». Розроблене додаток являє собою зручну модель для проведення експериментальних досліджень конкретної реалізації криптосистеми.

Були проведені експериментальні дослідження розробленої системи, в результаті яких була практично доведена статистична надійність основи ключа, генерованого математичною моделлю, за такими критеріями, як рівномірність розподілу випадкових чисел, їх повторюваність на різних відрізках згенерованої вибірки, а також повторюваність комбінацій сусідніх чисел.

Був проведений статистичний аналіз шифрувальної послідовності, що генерується самої криптосистемой на базі результатів роботи генератора хаосу, в результаті якого була доведена статистична надійність ключа і зашифрованих даних в бінарному, так і в числовому вигляді.

Був проведений тест стабільності роботи реалізованої математичної моделі генератора хаосу і самої розробленої криптосистеми. За результатами

тесту відхилення від очікувань виявилися дуже малі і, враховуючи досить велику довжину тестових даних, подібними відхиленнями можна знехтувати.

## Список використаної літератури


1. Мао Ст. Сучасна криптографія: Теорія і практика / В. Мао. — М.: Вільямс, 2005. — 768 с.
2. Барічев с.г. основи сучасної криптографії /с. г. Баричев, В. В. Гончаров, Р. є. Сєров. — М.: Гаряча лінія — Телеком, 2002. — 175 с.
3. Клод Шеннон. Теорія зв'язку в секретних системах / пер. з англ. В. Ф. Писаренко / / роботи з теорії інформації та кібернетики / за редакцією Р.Л. Добрушина та О. Б. Лупанова. — М: Видавництво іноземної літератури, 1963. — 829 с.
4. Нільс Фергюсон, Брюс Шнайер. Практична криптографія / Пер.з англ. Н. Н. Селіна, під редакцією А. В. Журавльова. — М.: Діалектика, 2004. — 432 с.
5. Ганзина Д. І. Застосування криптосистеми на генераторі хаосу «схема Чуа» для захисту інформації / Д. І. Ганзина, Т. А. Левицька // Збірник тез доповідей v Міжнародної науково-технічної конференції молодих вчених та студентів „Актуальні задачі сучасних технологій“, 17-18 листопада 2016 року. — Т.: ТНТУ, 2016. - Томіі. - С. 21.
6. Левицька, Т. А. Розробка криптосистеми на генераторі хаосу «схема Чуа» / Т. А. Левицька, Д. І. Ганзина // Університетська наука - 2017 : Міжнар. науково-техн. конф. (Маріуполь, 18-19 травня 2017 р.) : тез. докл. : в 3 т. / ДВНЗ "ПДТУ". - Маріуполь, 2017. - Т. 2. – С. 194-195.
7. Levitskaya, T. O. Data protection by using the «chua's circuit» chaos generator / T. O Levitskaya, D. I. Ganzina // Вісник Приазовського державного технічного університету : зб. наукових праць / ПДТУ. – Маріуполь, 2017. – Вип. 34. - С. 169-175.

8. Агурєєв К. І. застосування детермінованого хаосу для передачі інформації / Агурєєв К. І. / / Известия Тульського державного університету. Технічні науки. – 2014. - №11. - С. 197-212.

9. Когай Г. Д. методи і моделі хаотичних процесів в системах зв'язку / Г. Д. Когай, Т. Л. Тен. // Сучасні наукомісткі технології. – 2014. – № 10 – С. 61-64

10. Кузнєцов С. П. Динамічний хаос (курс лекцій). — М: Физматлит, 2001.

11. Лоренц Е. детермінований неперіодичний рух / / дивні атрактори. — М., 1981. — С. 88-116.

12. Стаття A Steganography Telecom System using a Chua Circuit Chaotic Noise Generator for data cryptography Apostolos P. Leros (1,2) and Antonios S. Andreatos Chaotic Modeling and Simulation (CMSIM) 1: 199-208, 2013

13. Пат. 7587047 США. Chaos generator for accumulation of stream entropy / Richard E. Crandall, Douglas P. Mitchell, Scott Krueger, Guy Tribble; Заявлено 22.06.09; Опубл. 08.09.09. – 3 с.

14. Булюк, І. Криптосистема зв'язку на схемі Чуа / І. Булюк, Я. Матвійчук // Теоретична електротехніка : зб. наука. пр. / Львів. держ. ун-т; редкол.: П. Г. Стахів (відп. ред.) та ін. — Л.: ЛНУ, 2008. — С. 48-51.

15. Галюк, с. Використання режиму узагальненної синхронізації при прихованій передачі інформації / С. Галюк, М. Кушнір, Д. Вовчук // Матеріали 5-ої Міжнародної конференції «Комп'ютерні науки та інженерія 2011». 24-26 листопада 2011-Львів.-С. 288-289.

16. ГОСТ34.602-89-Технічне завдання на створення автоматизованої системи. [Чинний від 1990]. 11 с. – (Національний стандарт України).

17. Кузнєцов А. П. Наочні образи хаосу /А. П. Кузнєцов //Соросівський освітній журнал. – 2000. – № 11. – С. 104-110.

18. Бугаєвський М. Ю. Дослідження поведінки ланцюга Чуа. Навчально-методичний посібник /М. Ю. Бугаєвський, в. І. Пономаренко. - Саратов: Видавництво ГосУНЦ «Коледж", 1998. — 29 с.

19. А. С. Городецький. Мінімальні атрактори і частково гіперболічні множини динамічних систем. Дисс. к. ф.-м. н., МДУ, 2001.

20. Грейди Буч. Мова UML. Посібник користувача = the Unified Modeling Language user guide. — 2-е вид. / Грейді Буч, Джеймс Рамбо, Айвар Джекобсон. — М., СПб.: ДМК прес, Пітер, 2004. — 432 с.

21. Білоусов а. Алгебра логіки і цифрові комп'ютери [Електронний ресурс] / А. Білоусов / / ALGLIB. – Режим доступу: http://alglib.sources.ru/articles/logic.php/.

22. Runge-Kutta Methods [Електроний ресурс]. Режим доступу: http://web.mit.edu/10.001/Web/Course_Notes/Differential_Equations_Notes/node5.html (дата звернення: 16.12.2017).

23. Rukhin, A., Soto, J., Nechvatal, J., Smid, M., Barker, E., Leigh, S., Levenson, M., Van- gel, M., Banks, D., Heckert, A., Dray, J., Vo, S. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications [Електроний ресурс]. Режим доступу: http://csrc.nist.gov/publications/nistpubs/800-22-rev1a/SP800-22rev1a.pdf (дата звернення 17.12.2017).

24. Brown R. Dieharder: A Random Number Test Suite [Електроний ресурс]. - Режим доступп: http://www.phy.duke.edu/ ~rgb/General/dieharder.php (дата звернення: 17.12.2017).

# ДОДАТОК 1

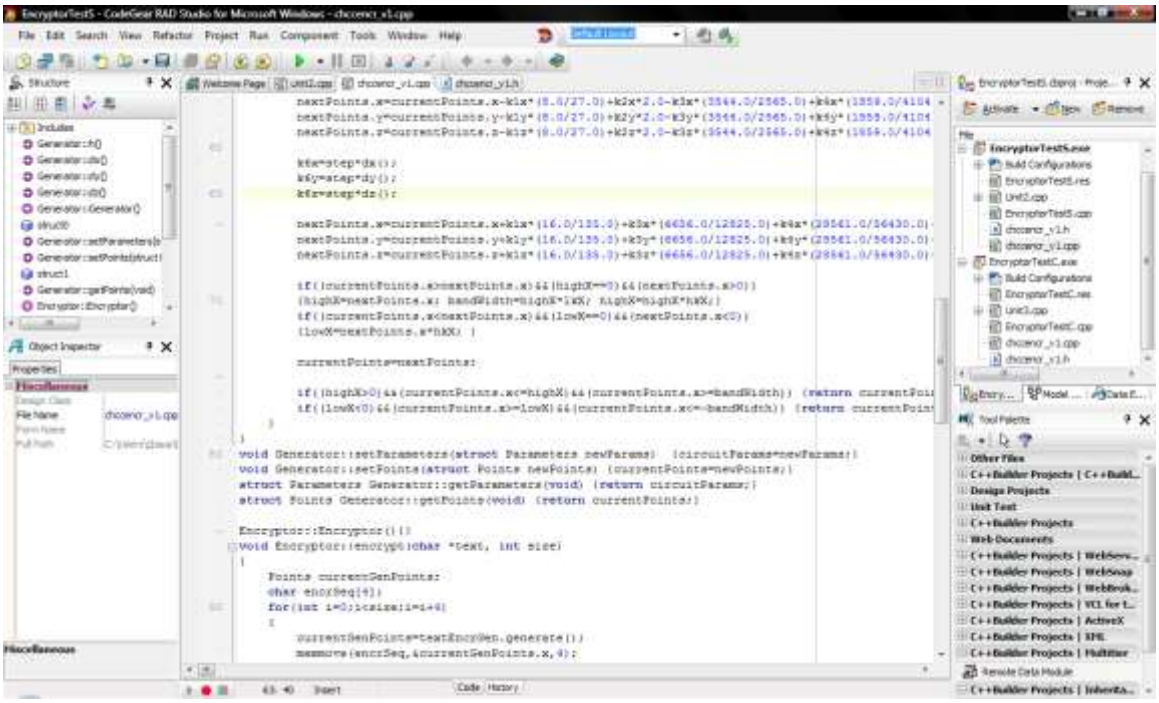

Рисунок 11 – Фрагмент програмного коду криптосистеми

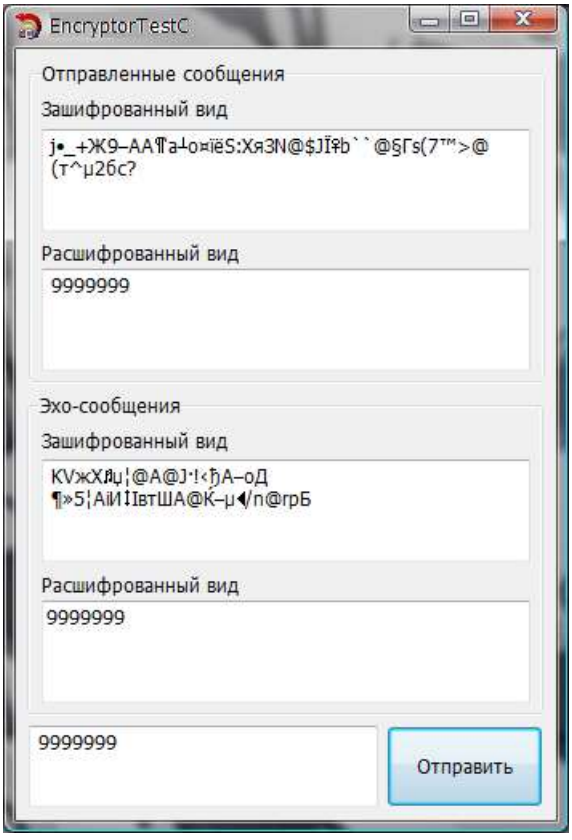

Рисунок 12 – Вікно провідного процесу

## ДОДАТОК 2

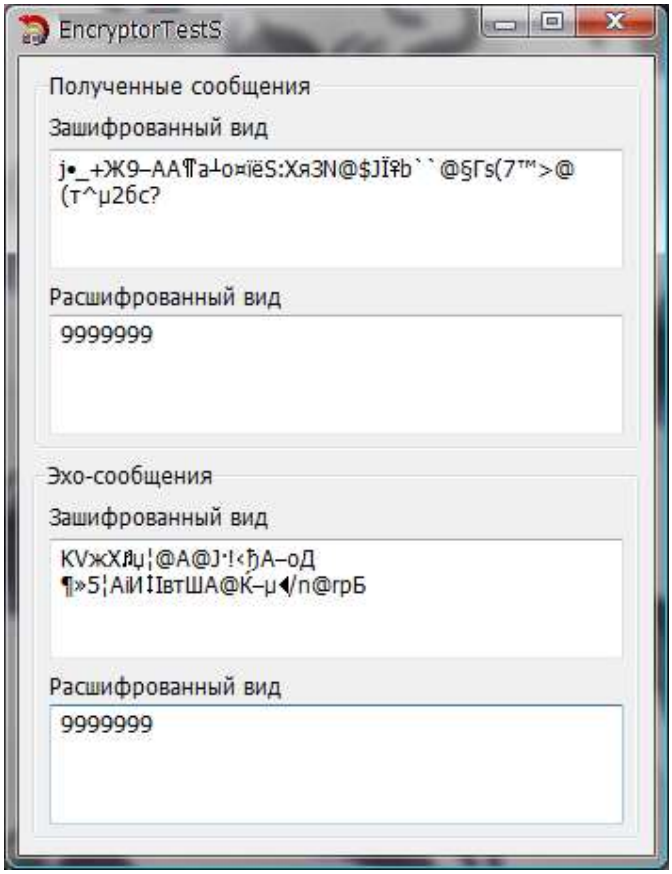


Рисунок 13 – Вікно веденого процесу

# ДОДАТОК 3

Таблиця 1 – Результати тестування ключа криптосистеми

| № | Назва тесту | Значення | Пройден |
|---|---|---|---|
| 1 | Дні народження (Birthday Spacings) | 0,486107 | Так |
| 2 | Пересічні перестановки (Overlapping Permutations) | 0,053402 | Так |
| 3 | Ранги матриць (Ranks of matrices) | 0,287288 | Так |
| 4 | Поток бітів (The bitstream test) | 0,978691 | Ні |
| 5 | Мавпячі тести (Monkey Tests) | 0,773721 | Так |
| 6 | Підрахунок одиниць (Count the 1’s) | 0,595824 | Так |
| 7 | Тест на паркування (Parking Lot Test) | 0,192328 | Так |
| 8 | Тест на мінімальну відстань (Minimum Distance Test) | 0,126506 | Так |
| 9 | Тест випадкових сфер (Random Spheres Test) | 0,254942 | Так |
| 10 | Тест стиснення (The Squeeze Test) | 0,981473 | Ні |
| 11 | Тест пересічних сум (Overlapping Sums Test) | 0,695497 | Так |
| 12 | Тест послідовностей (Runs Test) | 0,568498 | Так |
| 13 | Тест гри в кості (The Craps Test) | 0,373548 | Так |

# ДОДАТОК 4

Таблиця 2 – Результати тестування зашифрованих даних

| № | Название теста | Значение | Пройден |
|---|---|---|---|
| 1 | Дні народження (Birthday Spacings) | 0,398265 | Так |
| 2 | Пересічні перестановки (Overlapping Permutations) | 0,062948 | Так |
| 3 | Ранги матриць (Ranks of matrices) | 0,782236 | Так |
| 4 | Поток бітів (The bitstream test) | 0,974799 | Так |
| 5 | Мавпячі тести (Monkey Tests) | 0,753937 | Так |
| 6 | Підрахунок одиниць (Count the 1’s) | 0,478527 | Так |
| 7 | Тест на паркування (Parking Lot Test) | 0,187345 | Так |
| 8 | Тест на мінімальну відстань (Minimum Distance Test) | 0,113838 | Так |
| 9 | Тест випадкових сфер (Random Spheres Test) | 0,023938 | Ні |
| 10 | Тест стиснення (The Squeeze Test) | 0,942387 | Так |
| 11 | Тест пересічних сум (Overlapping Sums Test) | 0,284744 | Так |
| 12 | Тест послідовностей (Runs Test) | 0,533445 | Так |
| 13 | Тест гри в кості (The Craps Test) | 0,453874 | Так |