НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

ПОГОДЖЕНО	**ДОПУСКАЄТЬСЯ ДО ЗАХИСТУ**
Декан факультету	Завідувач кафедри
Інформаційних технологій	Комп'ютерних систем, мереж та кібербезпеки
_____ Болбот І.М., д.тех.н, проф.	_____ Касаткін Д.Ю., к. пед.н., доц.
підпис ПІБ, вчене звання і ступінь	підпис ПІБ, вчене звання і ступінь
«___»_____ 2025 р.	«___»_____ 2025 р.

КВАЛІФІКАЦІЙНА БАКАЛАВРСЬКА РОБОТА

На тему: «Розробка пристрою тестування псевдовипадкових послідовностей для систем захисту інформації»

Спеціальність 123 «Комп’ютерна інженерія»

Гарант освітньої програми:_____

Керівник дипломного проекту:_____ / Кулініч О.М. /
підпис ПІБ

Виконав:_____ / Сидоренко Д.Ю./
підпис ПІБ

КИЇВ-2025

НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ

«ЗАТВЕРДЖУЮ»
завідувач кафедри
комп'ютерних систем, мереж та кібербезпеки
_____/ Касаткін Д.Ю., к.п.н., доц. /
підпис ПІБ, вчене звання і ступінь
«__» _____ 20__ р.

З А В Д А Н Н Я

ДО ВИКОНАННЯ КВАЛІФІКАЦІЙНОЇ БАКАЛАВРСЬКОЇ СТУДЕНТУ

Дмитрій Юрійович Сидоренко
(прізвище, ім'я, по батькові)
Спеціальність (напрям підготовки): комп'ютерна інженерія_____

Тема кваліфікаційної бакалаврської роботи: «Розробка пристрою тестування псевдовипадкових послідовностей для систем захисту інформації»_____
_____
затверджена наказом ректора НУБіП України від "___" _____202_р. №____
Термін подання завершеної роботи на кафедру _____
Вихідні дані до кваліфікаційної бакалаврської роботи _____
_____
_____
_____
_____
Перелік питань, що підлягають розробці:
1. _____
2. _____
3. _____
Перелік графічного матеріалу (за потреби) _____
_____

Дата видачі завдання "_____"_____2024 р.

Керівник бакалаврської роботи _____ _/ Кулініч О.М.., д.т.н.
(підпис) (прізвище та ініціали)
Завдання прийняв до виконання _____ _/ Сидоренко Д.Ю._
(підпис) (прізвище та ініціали студента)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітка
1	Аналіз предметної області	04.02.2025 р.	Виконано
2	Проектування системи	00.00.2025 р.	Виконано
3	Реалізація системи	00.00.2025 р.	Виконано
4	Тестування системи	00.00.2025 р.	Виконано
5	Оформлення пояснювальної записки	00.00.2025 р.	Виконано
6	Оформлення графічного матеріалу	00.00.2025 р.	Виконано

Студент _____ **Сидоренко Д.Ю.**
(підпис) (ініціали та прізвище)

Керівник проекту (роботи) _____ **Олег Кулініч**
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Пояснювальна записка: сторінки 64, рисунки 4, таблиці 4, діаграми 2, додатки, джерел 17.

ПСЕВДОВИПАДКОВІ ПОСЛІДОВНОСТІ, АПАРАТНИЙ ПРИСТРІЙ, КРИПТОГРАФІЯ, ТЕСТУВАННЯ, NIST SP 800-22.

Об'єкт дослідження — пристрій для тестування псевдовипадкових послідовностей.

Метою роботи є розробка спеціалізованого апаратного пристрою для тестування псевдовипадкових послідовностей відповідно до сучасних вимог криптографічної безпеки, а також дослідження існуючих методів, стандартів тестування та реалізація їх у апаратному середовищі.

Перший розділ присвячено теоретичним основам псевдовипадкових послідовностей, аналізу їх властивостей, класифікації генераторів та стандартам тестування, зокрема NIST SP 800-22.

У другому розділі сформульовано вимоги до апаратного пристрою, побудовано його структурну та функціональну схеми, виконано обґрунтування вибору апаратного забезпечення, розроблено алгоритми тестування та побудовано блок-схеми роботи.

Третій розділ присвячено реалізації апаратної частини пристрою та програмного забезпечення для тестування ПВП, проведено експериментальні випробування, аналіз точності, надійності та ефективності тестів. Наведено результати тестування та їх порівняння з існуючими еталонними системами.

У результаті виконання дипломної роботи розроблено ефективний пристрій для тестування псевдовипадкових послідовностей, що підвищує надійність криптографічних систем, забезпечуючи контроль за якістю випадковості, необхідної для захисту інформації.

ЗМІСТ

ВСТУП

Сучасні інформаційні системи вимагають високого рівня захисту даних, особливо в умовах стрімкого розвитку інформаційних технологій і кіберзагроз. Одним із ключових компонентів криптографічного захисту є псевдовипадкові послідовності (ПВП), що використовуються у генераторах ключів, одноразових блокнотах, алгоритмах шифрування тощо. Від якості ПВП залежить надійність усієї криптосистеми.

ОскількиПВПєдетермінованимипослідовностями,сформованимизапевнималгоритмом,надзвичайноважливозабезпечитивідповідністьїхстатистичних характеристиктим,яківластивісправжнімвипадковимпослідовностям.Цьогодосягаютьшляхомтестуванняпвпзвикористаннямспеціалізованихстатистичних методів,якіявляютьаномаліїтанизькуентропійність.Середнихнайбільшупопулярністьздобулитестизnist,DiehardтаENT.

Незважаючи на наявність програмних інструментів для тестування ПВП, у ряді випадків виникає потреба у спеціалізованому апаратному рішенні, яке може здійснювати автономне тестування у режимі реального часу, зокрема, у вбудованих системах, пристроях шифрування або системах генерації ключів. Таким чином, виникає актуальне завдання створення пристрою, що здатен реалізовувати тестування ПВП на апаратному рівні.

Мета роботи – розробити пристрій для тестування псевдовипадкових послідовностей, який може бути використаний у системах захисту інформації.

Завдання роботи:

провести аналіз методів генерації та тестування ПВП;

визначити вимоги до апаратного пристрою тестування;

розробити архітектуру пристрою;

реалізувати прототип із вбудованим програмним забезпеченням;

провести тестування та аналіз результатів роботи пристрою.

Об’єкт дослідження – пристрої та методи тестування псевдовипадкових послідовностей.

Предмет дослідження – апаратні та програмні засоби, призначені для тестування ПВП у системах інформаційної безпеки.

Практична значущість розробки полягає у створенні функціонального пристрою, що може бути використаний як складова частина криптографічних модулів, вбудованих систем або тестувальних комплексів для оцінки генераторів ПВП.

РОЗДІЛ 1. ТЕОРЕТИЧНІ ОСНОВИ ПСЕВДОВИПАДКОВИХ ПОСЛІДОВНОСТЕЙ

1.1. Історія та розвиток псевдовипадкових послідовностей у криптографії

Ідея впровадження випадковості у криптографію має коріння в сиву давнину. Ще в античні часи шифри, як-от скітала чи шифр Цезаря, використовували компоненти, що визначалися випадковим вибором, наприклад, зміщення літер. Проте більш організоване застосування випадкових послідовностей виникло в двадцятому столітті разом із появою складніших шифрувальних пристроїв та прогресом математичної теорії інформації.

Одним з найперших прикладів використання справжньої випадковості є одноразовий блокнот, який теоретично гарантує абсолютну криптографічну безпеку. Цей метод було запропоновано на початку двадцятого століття та використовується до сьогодні в особливо критичних галузях, наприклад, у дипломатії чи спеціальних службах. У цій системі ключ повинен бути тієї ж довжини, що й повідомлення, і генерується повністю випадково. Основною трудністю такого підходу є складнощі з генерацією та збереженням справжніх випадкових послідовностей, особливо у великих масштабах.

З розвитком електроніки й комп'ютерів виникла потреба у відтворюваних послідовностях, які за зовнішніми ознаками виглядають випадковими, але можуть бути отримані повторно при наявності початкових даних. Такі послідовності отримали назву псевдовипадкових. У 1949 році Клод Шеннон, один із основоположників сучасної криптографії, у своїй роботі *"Communication Theory of Secrecy Systems"* заклав теоретичні основи захисту інформації, де важливу роль відіграють ймовірнісні процеси.

У 1960-х роках поширеними стали лінійні конгруентні генератори (ЛКГ), що мали форму:

$X_{n+1}=(aX_n+c) \mod m$ $X_{n+1} = (a X_n + c) \mod m$

Такі генератори широко застосовувалися у програмуванні та симуляціях, однак для криптографії виявилися занадто передбачуваними.

У 1970–80-х роках інтенсивно розроблялись регістри зсуву з лінійним зворотним зв'язком (LFSR), що поєднували простоту реалізації й добру статистичну якість. Їх почали вбудовувати в апаратні шифратори (наприклад, A5/1 у GSM).

Однак і LFSR мають слабкості, зокрема лінійність, що робить їх вразливими до аналізу. Це стимулювало розробку криптографічних генераторів ПВП**,** що базуються на нелінійних функціях, хешуваннях, блочних шифрах або потокових шифрах. У 1990-х роках з'явилися такі генератори, як Yarrow, Fortuna (від Брюса Шнайєра), а також стандарти від NIST, зокрема SP 800-90A, що описує генератори на основі HMAC, хешів та шифрування.

З кінця 2000-х років усе більшого значення набуває концепція CSPRNG (Cryptographically Secure Pseudorandom Number Generator) — криптографічно стійких генераторів, які мають гарантувати не лише хороші статистичні властивості, а й стійкість до атаки навіть при частковому компрометуванні внутрішнього стану генератора.

У сучасних інформаційних системах ПВП використовуються повсюдно: у криптографії, цифрових підписах, автентифікації, токенах, одноразових паролях (OTP), протоколах обміну ключами, а також у симуляціях та тестуванні.

Така історична еволюція показує, що вимоги до псевдовипадковості зростають разом із рівнем загроз. Саме тому актуальним завданням залишається розробка надійних засобів тестування таких послідовностей, зокрема в апаратному вигляді — для вбудованих систем, криптографічних пристроїв та IoT.

1.2. Поняття псевдовипадкових послідовностей і формальні критерії їх якості

Псевдовипадкові послідовності (ПВП) — це послідовності чисел або бітів, які генеруються детермінованими алгоритмами, але мають властивості, що наближаються до властивостей істинно випадкових послідовностей. Хоча вони не є справді випадковими, їхні характеристики дозволяють використовувати їх у багатьох сферах, де потрібна випадковість, зокрема в криптографії, моделюванні та статистичних дослідженнях.

Основою для конструювання ПВП виступають генератори псевдовипадкових чисел (PRNG), які, одержавши вхідне значення (насіння), породжують низку чисел, котра має вигляд випадкової. Такі генератори мусять гарантувати тривалий період повторюваності, однорідний розподіл та відсутність кореляцій між складовими послідовності.

У криптографії принципово важливо, щоб ПВП були непередбачуваними, без знання насіння. Тобто, навіть якщо відома частина вихідної послідовності, неможливо ефективно передбачити подальші елементи без знання початкового стану генератора. Ця властивість є критичною для забезпечення безпеки криптографічних систем.

Псевдовипадкові послідовності (ПВП) є базовим елементом у багатьох областях сучасних технологій, зокрема в криптографії та інформаційній безпеці. Хоча їх створюють детермінованими алгоритмами, їхні властивості наближаються до істинно випадкових послідовностей, що дозволяє їх ефективно використовувати в різних сферах.

Однією з ключових характеристик ПВП є детермінованість. Це означає, що за наявності конкретного вихідного значення (насіння) генератор завжди видає ту саму послідовність. Ця властивість забезпечує відтворюваність, що є корисною для тестування та налагодження систем. Проте, у контексті безпеки, детермінованість може бути вразливістю, якщо насіння стає відомим зловмисникові.

Іншою важливою властивістю є статистична випадковість. ПВП мають проходити різноманітні статистичні тести на випадковість, на кшталт тестів NIST SP 800-22, щоб гарантувати відсутність помітних

закономірностей. Це забезпечує, що послідовність не має передбачуваних шаблонів, які можуть бути використані для компрометації системи.

Непередбачуваність є найважливішою властивістю для криптографічних застосувань. У криптографії важливо, щоб без знання зерна було неможливо передбачити наступні елементи послідовності. Це забезпечує стійкість до атак, спрямованих на відновлення внутрішнього стану генератора або передбачення майбутніх значень.У практичному застосуванні ПВП використовуються для генерації ключів шифрування, ініціалізаційних векторів, сольових значень та інших криптографічних параметрів. Їх також застосовують у симуляціях, моделюванні, ігровій індустрії та машинному навчанні. У всіх цих випадках важливо забезпечити високу якість випадковості, щоб уникнути передбачуваності та забезпечити надійність систем.

Отже, псевдовипадкові послідовності є незамінним інструментом у сучасних технологіях, що гарантує необхідний рівень випадковості для різноманітних застосувань. Їх властивості, зокрема детермінованість, відтворюваність, статистична випадковість та непередбачуваність, роблять їх придатними для широкого спектра завдань, від криптографії до моделювання та аналізу даних.

Псевдовипадкові послідовності (ПВП) є фундаментальним елементом у багатьох сферах сучасних технологій, особливо в криптографії та інформаційній безпеці. Хоча вони створюються детермінованими алгоритмами, їхні властивості наближаються до істинно випадкових послідовностей, що дозволяє їх ефективно використовувати в різних галузях.

Однією з ключових характеристик ПВП є детермінованість. Це означає, що при заданому початковому значенні (насінні) генератор завжди видає ту саму послідовність. Ця властивість забезпечує відтворюваність, що є корисною для тестування та відлагодження систем. Проте, у контексті безпеки, детермінованість може бути вразливістю, якщо насіння стає відомим зловмиснику.

Ще однією важливою властивістю є статистична випадковість. ПВП повинні проходити різноманітні статистичні тести на випадковість, такі як тести NIST SP 800-22, щоб гарантувати відсутність помітних закономірностей. Це забезпечує, що послідовність не має передбачуваних шаблонів, які можуть бути використані для компрометації системи.

Непередбачуваність є критичною властивістю для криптографічних застосувань. У криптографії важливо, щоб без знання насіння було неможливо передбачити наступні елементи послідовності. Це забезпечує стійкість до атак, спрямованих на відновлення внутрішнього стану генератора або передбачення майбутніх значень.

У практичному застосуванні ПВП використовуються для генерації ключів шифрування, ініціалізаційних векторів, сольових значень та інших криптографічних параметрів. Їх також застосовують у симуляціях, моделюванні, ігровій індустрії та машинному навчанні. У всіх цих випадках важливо забезпечити високу якість випадковості, щоб уникнути передбачуваності та забезпечити надійність систем.

Таким чином, псевдовипадкові послідовності є незамінним інструментом у сучасних технологіях, забезпечуючи необхідний рівень випадковості для різноманітних застосувань. Їх властивості, такі як детермінованість, відтворюваність, статистична випадковість та непередбачуваність, роблять їх придатними для широкого спектра завдань, від криптографії до моделювання та аналізу даних.

Істинно випадкові послідовності генеруються з фізичних джерел ентропії, таких як атмосферний шум або радіоактивний розпад. Вони є непередбачуваними та не відтворюваними. Натомість ПВП є детермінованими та відтворюваними, але при цьому повинні бути статистично невідмінними від істинно випадкових послідовностей для забезпечення безпеки в криптографічних застосуваннях.

Псевдовипадкові послідовності (ПВП) є ключовим елементом у багатьох сферах сучасних технологій, де необхідна імітація випадковості.

Хоча вони створюються детермінованими алгоритмами, їхні властивості наближаються до істинно випадкових послідовностей, що дозволяє їх ефективно використовувати в різних галузях.

У криптографії ПВП використовуються для генерації ключів шифрування, ініціалізаційних векторів та сольових значень. Висока якість випадковості забезпечує стійкість до атак, оскільки передбачувані або повторювані послідовності можуть бути вразливими для криптоаналізу. Криптографічно стійкі псевдовипадкові генератори (CSPRNG) повинні відповідати суворим вимогам, включаючи непередбачуваність та проходження статистичних тестів, таких як NIST SP 800-22.

У симуляціях та моделюванні, особливо в методі Монте-Карло, ПВП дозволяють проводити численні експерименти з випадковими змінними для оцінки складних систем або процесів. Їх використання забезпечує відтворюваність результатів, що важливо для верифікації та валідації моделей.

В ігровій індустрії ПВП відповідають за генерацію випадкових подій, таких як випадкові зустрічі, розподіл ресурсів або поведінка неігрових персонажів. Це створює динамічний та непередбачуваний ігровий процес, що підвищує інтерес та залучення гравців.

У машинному навчанні ПВП використовуються для ініціалізації ваг нейронних мереж, випадкового розбиття даних на навчальні та тестові набори, а також у стохастичних алгоритмах оптимізації. Це сприяє уникненню переобучення та забезпечує кращу узагальнюваність моделей.

Крім того, ПВП застосовуються в телекомунікаціях для шифрування даних, у фінансовому моделюванні для оцінки ризиків, у біоінформатиці для моделювання генетичних процесів та в багатьох інших галузях, де необхідна імітація випадковості.

Псевдовипадкові послідовності є незамінним інструментом у сучасних технологіях, забезпечуючи необхідний рівень випадковості для різноманітних застосувань.

У криптографічних системах особливо важливо використовувати криптографічно стійкі псевдовипадкові генератори (CSPRNG), які забезпечують високий рівень непередбачуваності та стійкості до атак.

1.3. Класифікація генераторів псевдовипадкових послідовностей

Генератори псевдовипадкових послідовностей (ПВП), або PRNG (Pseudorandom Number Generators), — це програмні або апаратні засоби, призначені для створення числових чи бітових послідовностей, що імітують властивості справжньої випадковості. Залежно від принципу побудови, способу реалізації та області застосування, існує кілька типів генераторів, кожен з яких має свої переваги й обмеження.

Історично першими стали лінійні генератори, зокрема, лінійні конгруентні генератори (LCG), що генерують числа за простою формулою типу Xn+1=(aXn+c)mod mX_{n+1} = (aX_n + c) \mod m. Вони мають низьку обчислювальну складність і використовуються у багатьох прикладних задачах, але не підходять для криптографії через обмежений період і низьку непередбачуваність. Іншим популярним варіантом стали регістри зсуву з лінійним зворотним зв’язком (LFSR), особливо зручні для реалізації в апаратурі. Вони забезпечують хорошу швидкість, але вразливі до аналізу через лінійний характер.

Нелінійні генератори були створені для подолання недоліків попередніх. Прикладом є NLFSR — регістри з нелінійною функцією зворотного зв’язку, що значно ускладнюють передбачення наступних значень. Крім того, широко використовуються генератори, побудовані на криптографічних примітивах: хеш-функціях (SHA-2, SHA-3), блочних шифрах (AES у режимі лічильника), потокових шифрах (ChaCha20). Такі генератори мають високу стійкість до атак, що робить їх придатними для криптографічних застосувань.

За способом реалізації розрізняють програмні та апаратні генератори. Програмні реалізації зустрічаються у звичайних додатках, операційних системах та бібліотеках (наприклад, rand() у C або random() у Python). Апаратні генератори ґрунтуються на фізичних джерелах ентропії — шумах напруги, електромагнітних коливаннях, квантових ефектах. Вони дають по-справжньому випадкові значення, але є складними та дорогими у виробництві. Найчастіше в реальних системах використовується комбінований підхід: апаратний генератор використовується для ініціалізації програмного ПВП або як джерело ентропії для поновлення його внутрішнього стану.

Залежно від призначення, генератори поділяються на загального призначення та криптографічні. Перші застосовуються у симуляціях, статистичному моделюванні, ігровій індустрії. Вони не мають вимог до непередбачуваності, тому їх реалізація проста і швидка. Криптографічні генератори (CSPRNG) навпаки — повинні бути непередбачуваними навіть за часткової компрометації внутрішнього стану. Вони повинні задовольняти вимоги до forward secrecy та backtracking resistance. Їх застосовують у шифруванні, автентифікації, генерації ключів та протоколах обміну.

Також генератори поділяють за способом отримання початкових даних, або насіння (seed). Якщо використовується фіксоване насіння, то генерація є повністю детермінованою й відтворюваною — це корисно для тестування або симуляцій. Якщо ж насіння генерується із зовнішніх джерел ентропії (час, рух миші, шум процесора), то кожен запуск генератора забезпечує унікальну послідовність.

Розуміння типів генераторів і їх характеристик є критично важливим для створення надійних систем захисту інформації, де вибір ПВП напряму впливає на стійкість криптографічних алгоритмів. У межах цього проєкту головний фокус буде спрямовано на генератори, придатні до апаратної реалізації і такі, що забезпечують криптографічну стійкість.

1.4. Критерії оцінки якості псевдовипадкових послідовностей

Оцінка якості псевдовипадкових послідовностей (ПВП) є ключовим етапом у перевірці їх придатності до використання у системах захисту інформації. Незважаючи на те, що ПВП формуються детерміновано, вони повинні відповідати певним властивостям, притаманним істинно випадковим послідовностям. Якщо генератор не задовольняє цим властивостям, він може бути вразливим до статистичного або криптоаналітичного аналізу.

Першою і основною властивістю є статистична випадковість. Вона означає, що у довгій ПВП повинна бути приблизно рівна кількість нулів і одиниць, не повинно бути надто частих або регулярних шаблонів, а також не повинно спостерігатися надмірних серій однакових символів. Також важливо, щоб послідовність не демонструвала автокореляції — тобто, не мала передбачуваних залежностей між значеннями на різних відстанях. Важливо підкреслити, що ці властивості стосуються не лише загального вигляду послідовності, а й її поведінки у контексті великих обсягів даних.

Другою важливою характеристикою є періодичність. Ідеальна ПВП має максимально довгий період, після якого послідовність починає повторюватися. У генераторах, де період невеликий, можна передбачити наступні значення після накопичення певної кількості спостережень. У криптографічних генераторах період повинен бути щонайменше 21282^{128}, що практично виключає повторення послідовності за час роботи реальної системи.

Третім важливим критерієм є ентропія — міра невизначеності послідовності. В ідеальній ситуації кожен біт у ПВП несе один біт ентропії, тобто є абсолютно непередбачуваним. Якщо ж ентропія значно нижча, це свідчить про можливу детерміновану структуру або повторюваність, що знижує безпеку всієї криптосистеми. Ентропія вимірюється за допомогою математичних методів, зокрема аналізу частотних розподілів або стискання (чим краще послідовність стискається, тим нижча її ентропія).

Четвертий критерій — це непередбачуваність. Вона особливо важлива в контексті криптографії. Навіть якщо ПВП виглядає статистично випадковою, її значення можуть бути передбачуваними, якщо зловмисник отримає доступ до частини послідовності або до внутрішнього стану генератора. Тому криптографічні генератори повинні мати властивості forward secrecy (неможливість визначити наступні значення при знанні попередніх) та backward secrecy (неможливість відновити попередні значення при компрометації поточного стану).

Оцінювання якості ПВП здійснюється за допомогою різних методик і тестів. Найбільш відомим є стандарт NIST SP 800-22, що містить набір з понад 15 статистичних тестів, які дозволяють виявити відхилення від випадковості: частотний тест, тест на довжину серій, тест на відсутність шаблонів, тест на автокореляцію тощо. Іншими популярними наборами тестів є Diehard tests та ENT, які також забезпечують аналіз на предмет рівномірності, ентропії, серій та біасу (схилу до певних значень).

У практиці розробки систем інформаційної безпеки важливо не лише пройти такі тести, а й інтегрувати їх у процес контролю якості — наприклад, у вигляді апаратного або програмного модуля, який автоматично перевіряє ПВП на відповідність стандарту перед використанням у шифруванні. Саме такий підхід і буде реалізований у межах цього дипломного проєкту — створення пристрою, що здатен виконувати базові статистичні тести ПВП в реальному часі.

1.5. Стандарти тестування псевдовипадкових послідовностей

Щоб забезпечити відповідність псевдовипадкових послідовностей вимогам до криптографічної надійності, необхідно використовувати стандартизовані методи їх тестування. Такі методи дозволяють виявити наявність шаблонів, статистичних відхилень, низької ентропії або інших небажаних властивостей, які можуть знизити безпеку системи. Протягом

останніх десятиліть було створено низку міжнародних стандартів і тестових пакетів, що стали галузевими еталонами.

Найбільш відомим і поширеним стандартом є NIST SP 800-22, розроблений Національним інститутом стандартів і технологій США. У цьому стандарті представлено набір з 15 базових тестів, призначених для оцінки різних статистичних властивостей бітових послідовностей. Серед них — частотний тест (Frequency Test), який перевіряє баланс нулів і одиниць; тест довжин серій (Runs Test), що оцінює довжини підряд ідучих бітів одного значення; тест на лінійність; тест на автокореляцію; та інші. Кожен з тестів виводить p-значення, що визначає ймовірність того, що спостережуване відхилення від випадковості є результатом випадкового флуктуації, а не систематичної помилки.

Іншим важливим набором тестів є Diehard tests, розроблений Джорджем Марсальєю у 1990-х роках. Цей набір містить декілька складніших статистичних тестів, зокрема на відстань між рівними значеннями, на збіг бітових шаблонів, на незалежність чисел, що генеруються. Він більше орієнтований на числові послідовності, ніж на бітові, та застосовується у випадках, коли потрібно детально дослідити генератори з довгими періодами.

Ще один набір — ENT (A Pseudorandom Number Sequence Test Program) — є легким у реалізації та забезпечує базову оцінку ентропії, біасу, рівномірності та здатності до стискання. ENT широко використовується як швидкий тест при розробці простих генераторів.

Також існують більш розширені системи, як TestU01, що містить десятки тестів і дозволяє створювати комбіновані програми тестування. Цей набір використовується переважно для академічних досліджень та тестування генераторів, що працюють у надзвичайно великих обсягах даних.

Усі ці тести мають одну спільну особливість — вони не гарантують абсолютної випадковості, а лише перевіряють, чи немає ознак невипадковості. Результати тестів повинні тлумачитися статистично: навіть

справжня випадкова послідовність іноді може не пройти один із тестів. Тому важливо оцінювати результати у сукупності, на основі великої кількості запусків, та враховувати ймовірність хибних спрацьовувань.

У криптографії також застосовуються стандарти FIPS 140-2/3, які визначають вимоги до криптографічних модулів, включаючи процедури тестування генераторів випадкових чисел. Для генераторів, що претендують на сертифікацію, обов'язковою є реалізація внутрішнього самотестування, наприклад тест на повторення або тест на блокову ідентичність, які мають виконуватись у реальному часі.

У межах цієї роботи для реалізації тестування буде відібрано кілька найбільш інформативних і компактних з точки зору обчислювальної складності тестів із набору NIST і ENT, адаптованих для реалізації в апаратній системі з обмеженими ресурсами.

1.6. Приклади використання ПВП у системах захисту інформації

Псевдовипадкові послідовності (ПВП) є невід'ємною складовою сучасних систем захисту інформації. Вони забезпечують необхідний рівень випадковості та непередбачуваності, що є критичними для ефективного функціонування криптографічних та інформаційно-безпекових систем.

Криптографічно стійкі псевдовипадкові генератори (CSPRNG) повинні відповідати суворим вимогам, включаючи непередбачуваність та проходження статистичних тестів, таких як NIST SP 800-22. Цей стандарт містить набір з 15 статистичних тестів, розроблених для оцінки випадковості бінарних послідовностей, що генеруються апаратними або програмними засобами. Тести включають перевірку частоти одиниць і нулів, довжини серій, спектральний аналіз та інші методи, спрямовані на виявлення закономірностей, які можуть свідчити про невипадковість послідовності .

Використання CSPRNG, які успішно проходять ці тести, є критично важливим для забезпечення безпеки криптографічних систем. Наприклад, у протоколах шифрування, таких як TLS або IPsec, генерація ключів повинна бути максимально непередбачуваною, щоб запобігти можливості їх

відновлення зловмисниками. Також у системах автентифікації ПВП застосовуються для створення одноразових паролів (OTP), які використовуються лише один раз і мають обмежений час дії, що значно підвищує безпеку.

Крім того, ПВП використовуються в протоколах безпеки для генерації випадкових чисел, необхідних для створення сесійних ключів, сольових значень у хеш-функціях та інших криптографічних операцій. Це забезпечує унікальність та непередбачуваність кожної сесії, що ускладнює проведення атак типу "повторне відтворення" або "людина посередині".

ПВП також використовуються для маскування каналів передачі інформації. Це дозволяє приховати фактичну передачу даних, роблячи її невидимою для потенційного зловмисника. Генератори псевдовипадкових чисел можуть створювати шум, який маскує справжні сигнали, ускладнюючи їх виявлення та аналіз.

У системах автентифікації ПВП застосовуються для створення одноразових паролів (OTP), які використовуються лише один раз і мають обмежений час дії. Це значно підвищує безпеку, оскільки навіть у разі перехоплення пароля він не може бути повторно використаний.

Крім того, ПВП використовуються в протоколах безпеки для генерації випадкових чисел, необхідних для створення сесійних ключів, сольових значень у хеш-функціях та інших криптографічних операцій. Це забезпечує унікальність та непередбачуваність кожної сесії, що ускладнює проведення атак типу "повторне відтворення" або "людина посередині".

У сфері моделювання та симуляцій ПВП застосовуються для створення випадкових сценаріїв, що дозволяє тестувати системи безпеки в умовах, наближених до реальних. Це включає моделювання атак, тестування стійкості систем до різних загроз та оцінку ефективності заходів безпеки.

Таким чином, псевдовипадкові послідовності є незамінним інструментом у сучасних технологіях, забезпечуючи необхідний рівень випадковості для різноманітних застосувань. Їх властивості, такі як

детермінованість, відтворюваність, статистична випадковість та непередбачуваність, роблять їх придатними для широкого спектра завдань, від криптографії до моделювання та аналізу даних.

1.7. Аналіз сучасного стану задачі. Постановка задачі розробки пристрою

У сучасному світі інформаційної безпеки псевдовипадкові послідовності (ПВП) відіграють ключову роль у забезпеченні конфіденційності, цілісності та доступності даних. Вони використовуються в криптографічних протоколах, генерації ключів, автентифікації та інших критичних процесах. Однак, ефективність цих систем безпосередньо залежить від якості ПВП, що вимагає надійних методів їх тестування.

Одним із найвідоміших стандартів для оцінки випадковості ПВП є NIST SP 800-22, який містить 15 статистичних тестів, спрямованих на виявлення відхилень від ідеальної випадковості. Ці тести включають перевірку частоти, довжини серій, спектральний аналіз та інші методи. Проте, як зазначено в публікаціях, жоден набір тестів не може гарантувати абсолютну випадковість послідовності, і результати тестування слід інтерпретувати з урахуванням специфіки застосування.

Незважаючи на широке використання програмних засобів для тестування ПВП, існують певні обмеження. По-перше, програмне тестування часто вимагає значних обчислювальних ресурсів та часу, особливо при аналізі довгих послідовностей. По-друге, програмні засоби можуть бути вразливими до атак, спрямованих на компрометацію середовища виконання. По-третє, відсутність апаратної реалізації ускладнює інтеграцію тестування в реальному часі в системи з обмеженими ресурсами.

У зв'язку з цим виникає потреба в розробці спеціалізованого апаратного пристрою для тестування ПВП. Такий пристрій повинен забезпечити швидке та надійне визначення якості ПВП відповідно до встановлених стандартів. Апаратне тестування має кілька переваг: високу

швидкодію, надійність, можливість інтеграції в існуючі системи та автономність роботи.

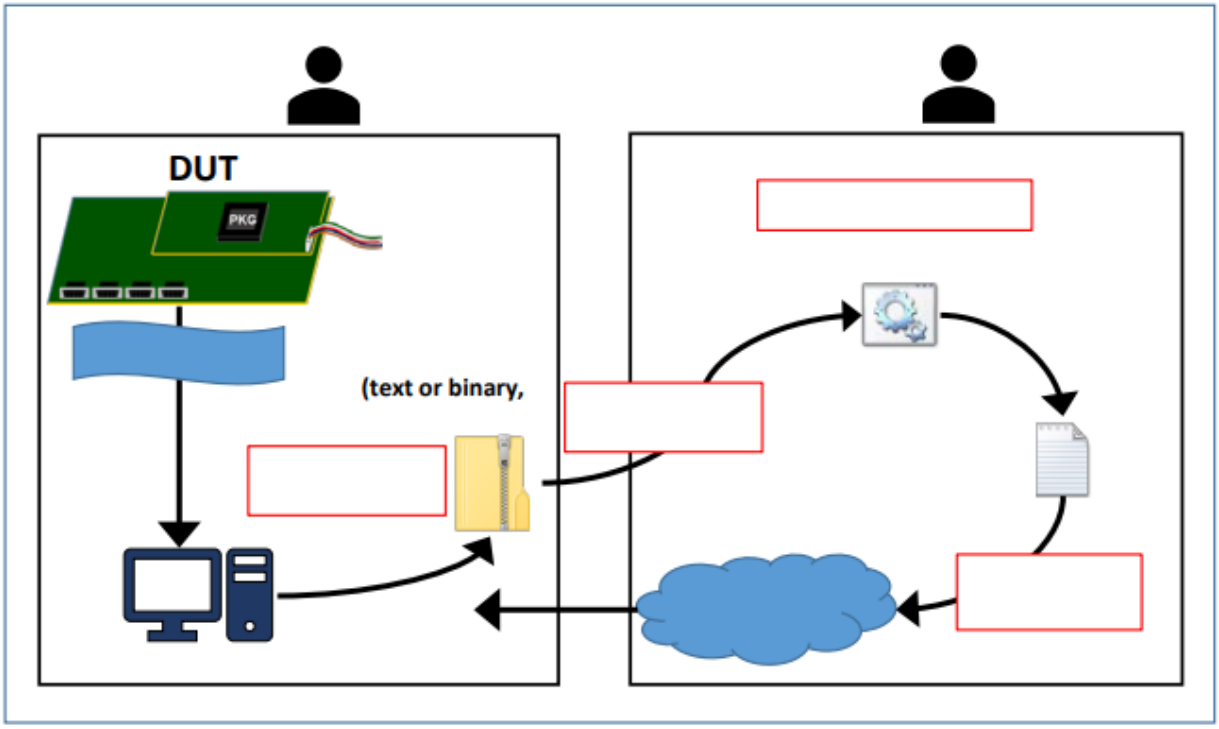

Рисунок 1.1 – Загальна схема взаємодії пристрою тестування ПВП з зовнішнім середовищем

Метою даної роботи є розробка спеціалізованого апаратного пристрою для тестування псевдовипадкових послідовностей. Основні завдання включають аналіз існуючих методів та стандартів тестування ПВП, розробку архітектури пристрою, реалізацію алгоритмів тестування в апаратному середовищі, проведення експериментального тестування та аналіз результатів.

Пристрій повинен відповідати наступним функціональним та технічним вимогам: підтримка стандартів (наприклад, NIST SP 800-22), висока швидкодія, модульність, наявність зручних інтерфейсів для введення даних та виведення результатів, а також оптимізація енергоспоживання для можливості використання в системах з обмеженими ресурсами.

Традиційно для оцінки якості ПВП використовуються програмні засоби, які реалізують стандартизовані тести, такі як NIST SP 800-22. Ці тести включають перевірку частоти, довжини серій, спектральний аналіз та

інші методи, спрямовані на виявлення відхилень від ідеальної випадковості. Проте, як зазначено в дослідженнях, жоден набір тестів не може гарантувати абсолютну випадковість послідовності, і результати тестування слід інтерпретувати з урахуванням специфіки застосування.

У цьому розділі було проаналізовано сучасний стан методів тестування псевдовипадкових послідовностей (ПВП) та обґрунтовано необхідність розробки спеціалізованого апаратного пристрою для їх оцінки. Існуючі програмні засоби, хоча й широко використовуються, мають обмеження щодо швидкодії, надійності та інтеграції в системи з обмеженими ресурсами. Розробка апаратного рішення дозволить забезпечити більш ефективне, надійне та автономне тестування ПВП, що є критично важливим для підвищення безпеки сучасних інформаційних систем.

Розділ 2. ПРОЄКТУВАННЯ ПРИСТРОЮ ТЕСТУВАННЯ ПВП

2.1. Формування вимог до пристрою

У процесі створення пристрою для тестування псевдовипадкових послідовностей (ПВП) першочерговим кроком є визначення чітких функціональних та технічних вимог до нього. Ці вимоги повинні забезпечувати відповідність пристрою стандартам оцінювання випадковості, його ефективність у роботі з великими обсягами даних, здатність до інтеграції в існуючі системи та зручність використання. Формування вимог базується як на аналізі сучасних викликів у сфері інформаційної безпеки, так і на технічних характеристиках стандартів, таких як NIST SP 800-22.

Основною функцією пристрою є тестування якості ПВП, що генеруються зовнішніми або вбудованими генераторами. Пристрій повинен мати здатність обробляти великі об’єми вхідних даних у реальному часі, виконуючи при цьому стандартні статистичні тести, включаючи, але не обмежуючись тестом на частоту, серії, інтервали, спектральний тест тощо. Для цього він повинен бути оснащений обчислювальним блоком з відповідною потужністю, здатним виконувати ці тести з мінімальними затримками.

Крім обчислювальної спроможності, важливим є наявність надійних інтерфейсів обміну даними — таких як USB, UART, SPI або Ethernet — для завантаження тестованих послідовностей та виведення результатів тестів. Перевагою буде підтримка як текстового, так і бінарного формату даних. Пристрій повинен мати можливість працювати як в автономному режимі, так і у складі комп’ютерної або вбудованої системи, що потребує гнучкої архітектури керування.

Зважаючи на потенційне використання пристрою у вбудованих системах з обмеженими ресурсами (наприклад, у пристроях IoT або мережевому обладнанні), важливо враховувати енергоспоживання. Пристрій

повинен бути енергоефективним та оптимізованим з точки зору витрат пам’яті, пропускної здатності каналів обміну та енергетичних характеристик.

Також серед вимог — модульність конструкції. Архітектура пристрою має дозволяти оновлення програмного забезпечення (наприклад, прошивки) та додавання нових тестів у майбутньому без необхідності повної реконструкції пристрою. Це забезпечить масштабованість та адаптивність рішення в умовах змін стандартів та нових вимог до безпеки.

З урахуванням наведеного, до пристрою пред’являються такі загальні вимоги: висока точність оцінки ПВП; відповідність існуючим стандартам; можливість швидкої обробки великого обсягу даних; автономна та інтегрована робота; підтримка оновлення прошивки; низьке енергоспоживання; наявність зрозумілого інтерфейсу для користувача.

Для реалізації пристрою доцільним є використання мікроконтролера серії **STM32F4**, який має високу обчислювальну потужність завдяки 32-бітному ядру ARM Cortex-M4, вбудованому апаратному множенню та модулю плаваючої коми. Він також підтримує роботу з зовнішньою пам’яттю та має достатньо портів введення/виведення для реалізації інтерфейсів, таких як UART, USB або SPI. Завдяки широкій екосистемі підтримки (STM32CubeIDE, бібліотеки HAL/LL), розробка програмного забезпечення є ефективною та стандартизованою. Водночас пристрій може працювати на базі **FreeRTOS** — реального часу операційної системи, що дозволяє паралельно обробляти декілька потоків даних, включаючи тестування, передачу результатів, логування та віддалене керування.

Використання STM32 дає змогу не лише забезпечити необхідну продуктивність, але й легко масштабувати проєкт для майбутніх оновлень чи ускладнень архітектури.

Отже, на цьому етапі важливо не лише визначити ключові функції пристрою, а й врахувати особливості його подальшої експлуатації, масштабування та адаптації до різних технічних умов. Вимоги, що були сформульовані, закладають основу для ефективного проєктування системи,

яка здатна забезпечити надійне тестування ПВП як у автономному режимі, так і в складі комплексних інформаційних систем.

2.2. Структурна та функціональна схема

Проєктування пристрою для тестування псевдовипадкових послідовностей вимагає чіткого визначення його структурної та функціональної організації. Структурна схема є концептуальним представленням взаємозв'язків між основними апаратними компонентами, тоді як функціональна схема описує розподіл завдань між окремими модулями пристрою та послідовність їхньої взаємодії.

У центрі структурної схеми пристрою тестування псевдовипадкових послідовностей розміщується мікроконтролер STM32F4, який виконує функції керуючого та обчислювального ядра. Він забезпечує координацію роботи всіх апаратних компонентів системи. До мікроконтролера під'єднано блок введення/виведення, що відповідає за обмін даними з користувачем або зовнішніми джерелами за допомогою інтерфейсів, таких як UART або USB. Для обробки даних у режимі реального часу використовується оперативна пам'ять типу SRAM, яка тимчасово зберігає вхідну послідовність та проміжні результати обчислень. Постійна пам'ять FLASH використовується для зберігання прошивки пристрою, реалізації в ній алгоритмів статистичного тестування та збереження налаштувань системи. Обчислювальні процедури, пов'язані з виконанням тестів випадковості (в тому числі за методиками NIST SP 800-22), реалізуються в основному обчислювальному блоці, що функціонує у межах апаратної платформи. Результати аналізу можуть виводитися через інтерфейс користувача, до якого можуть входити дисплей, світлодіоди або веб-панель залежно від конфігурації. Для синхронного виконання задач збору, обробки та передачі даних у пристрої використовується легка операційна система реального часу, зокрема FreeRTOS, яка дозволяє ефективно розподіляти ресурси системи між паралельними процесами. Така архітектура дозволяє забезпечити надійну та

гнучку роботу пристрою, що відповідає вимогам до сучасних систем інформаційної безпеки.

На рівні функціональної взаємодії пристрій працює за наступним принципом. Користувач передає на вхід пристрою бітову послідовність або файл із нею. Вона завантажується в оперативну пам’ять і обробляється обчислювальним блоком, який послідовно виконує задані тести. Після кожного тесту результат зберігається або виводиться в інтерфейс користувача, а також, за необхідності, передається через комунікаційний порт на комп’ютер для подальшого аналізу чи зберігання.

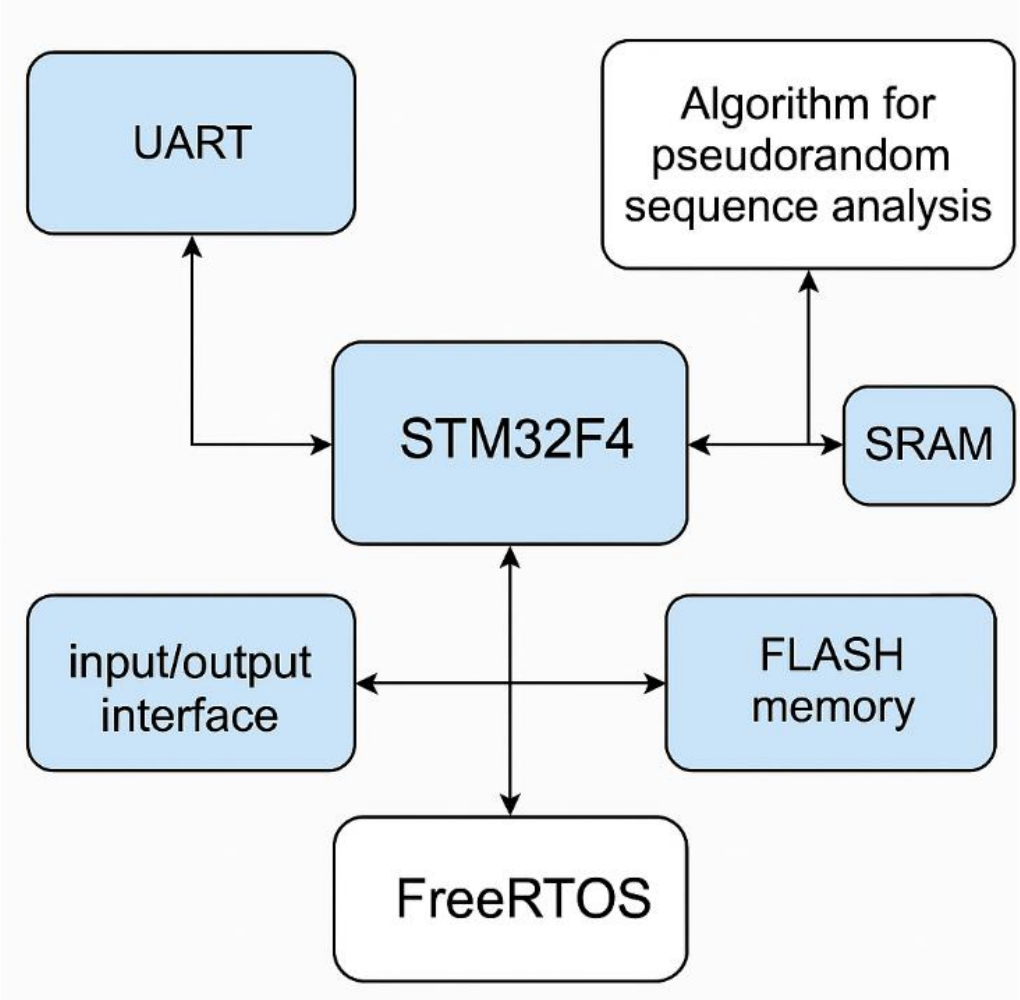

Рисунок 2.1 – Структурна схема пристрою тестування ПВП на базі STM32F4

У підсумку, запропонована структурна та функціональна організація пристрою забезпечує не лише ефективність виконання алгоритмів тестування, але й високий ступінь адаптивності до різних сценаріїв використання. Завдяки модульному підходу, пристрій легко інтегрується в

існуючі інформаційні системи, а його енергоефективність та обчислювальна гнучкість відповідають актуальним викликам у сфері безпеки та вбудованих технологій.

2.3. Вибір апаратного забезпечення

На етапі розробки пристрою тестування псевдовипадкових послідовностей (ПВП) надзвичайно важливим є обґрунтований вибір апаратної платформи, яка б забезпечувала необхідну продуктивність, енергоефективність, масштабованість і стабільність роботи у цільових умовах. Зважаючи на потребу виконання обчислювально складних статистичних тестів (зокрема, згідно зі стандартом NIST SP 800-22), критичним є наявність потужного обчислювального ядра, достатнього обсягу пам’яті та швидкодійних інтерфейсів обміну даними.

Для реалізації проєкту було обрано мікроконтролер STM32F407VG з ядром ARM Cortex-M4, який поєднує високу обчислювальну потужність, апаратне прискорення операцій із плаваючою комою (FPU) та підтримку великої кількості периферій. Таке рішення забезпечує належну швидкодію при виконанні математичних обчислень, необхідних для реалізації алгоритмів аналізу ПВП. Крім того, даний контролер підтримує зовнішню оперативну пам’ять (SRAM), має до 1 МБ FLASH-пам’яті та 192 КБ вбудованої оперативної пам’яті, що дозволяє реалізувати більшість статистичних тестів без потреби у додаткових зовнішніх модулях зберігання даних.

Щодо комунікації з користувачем або зовнішніми системами, STM32F407 має вбудовані інтерфейси USB, UART, SPI, I2C, CAN, які забезпечують гнучкі варіанти інтеграції. У контексті нашого пристрою основним каналом обміну є UART, який дозволяє передавати як вхідні послідовності для тестування, так і результати перевірки. За потреби можливе підключення дисплея або сенсорного екрана для локального виводу результатів і конфігурації.

Для збереження енергонезалежної інформації та оновлення прошивки пристрою передбачено використання вбудованої FLASH-пам’яті, а також можлива реалізація microSD-слоту для зберігання великих об’ємів послідовностей або логів тестування. В якості джерела живлення передбачено використання стандартного USB-порту з підтримкою напруги 5В, що дозволяє експлуатацію пристрою як у лабораторних, так і в польових умовах.

Компонент	**Характеристика**
Мікроконтролер STM32F407VG	ARM Cortex-M4, 168 МГц, FPU, 100+ GPIO
FLASH-пам’ять (вбудована)	1 МБ вбудованої пам’яті для прошивки та даних
Оперативна пам’ять (SRAM)	192 КБ внутрішньої пам’яті + можливість підключення зовнішньої
Інтерфейси зв’язку	UART, USB, SPI, I2C, CAN
Дисплей (опціонально)	OLED/TFT 0.96–2.8” з інтерфейсом SPI або I2C
Індикатори	Світлодіодні індикатори (статус, помилки, завершення)
Операційна система	FreeRTOS (управління задачами та чергами)
Живлення	USB 5 В або зовнішнє живлення через стабілізатор

Таблиця 2.1 – Технічні характеристики основних апаратних компонентів пристрою тестування ПВП

Для індикації стану пристрою обрано світлодіодні індикатори, які сигналізують про початок тестування, помилки, завершення аналізу, а також режим очікування або завантаження. У разі потреби система може бути розширена за рахунок підключення дисплея OLED чи TFT, які забезпечують вивід інформації в більш зручному форматі.

Додатково, для організації внутрішньої логіки та паралельної обробки задач, планується використання вбудованої операційної системи FreeRTOS, яка надає можливість гнучко управляти задачами, таймерами, чергами та міжпроцесною взаємодією без перевантаження апаратної частини.

Загалом, обране апаратне рішення на основі STM32F4 є технічно обґрунтованим, з огляду на баланс між ціною, функціональністю та можливістю реалізації алгоритмів тестування ПВП. Воно дозволяє ефективно реалізувати поставлені задачі в межах заданих ресурсних обмежень, при цьому забезпечуючи можливість подальшого розширення функціоналу пристрою.

2.4. Розробка алгоритмів тестування ПВП

У процесі побудови пристрою для тестування псевдовипадкових послідовностей (ПВП) одним із ключових етапів є створення ефективних алгоритмів перевірки, що дозволяють оцінити рівень випадковості, ентропійності та криптографічної стійкості таких послідовностей. Ці алгоритми мають забезпечити виявлення закономірностей, повторів або статистичних відхилень, які можуть свідчити про знижену якість генерації та, як наслідок, про зниження рівня захищеності інформаційної системи.

Постановка задачі

Алгоритми тестування псевдовипадкових послідовностей мають забезпечувати комплексну перевірку якості згенерованих бітових потоків. Їхнє головне завдання полягає у виявленні статистичних і структурних відхилень, які можуть свідчити про зниження рівня випадковості. Зокрема, вони повинні виявляти відхилення від рівномірного розподілу нулів та одиниць, перевіряти наявність або відсутність кореляцій між сусідніми елементами послідовності, а також визначати ознаки періодичності, які можуть вказувати на регулярну або повторювану природу сигналу. Важливою складовою перевірки є здатність виявити надмірну регулярність, яка часто свідчить про недосконаліть генератора, та оцінити рівень

інформаційної ентропії послідовності — одного з базових критеріїв криптографічної стійкості. Ефективні алгоритми тестування мають поєднувати ці функції в єдиному процесі обробки з можливістю адаптації до обсягу даних та умов експлуатації пристрою.

Розроблені алгоритми повинні працювати в режимі обробки великих обсягів бітової інформації, підтримувати автоматизований аналіз результатів і надавати кількісні оцінки якості ПВП.

Основні типи алгоритмів тестування

У межах цієї роботи були реалізовані такі класи алгоритмів:

1. Частотний тест (Frequency test)

Це один із найпростіших, але водночас ефективних тестів. Його мета полягає в перевірці гіпотези про рівномірний розподіл нулів та одиниць у ПВП. Алгоритм передбачає підрахунок кількості кожного з бітів і перевірку відхилення від ідеального розподілу (тобто 50/50) за допомогою критеріїв, таких як χ^2 або нормоване відхилення. Формула для обчислення нормалізованого статистичного показника виглядає наступним чином:

S=|n1−n0|nS = \frac{|n_1 - n_0|}{\sqrt{n}}

де n1n_1 – кількість одиниць, n0n_0 – кількість нулів, n=n1+n0n = n_1 + n_0 – загальна довжина послідовності.

2. Тест серій (Runs test)

Цей тест виявляє надмірну або недостатню регулярність у чергуванні бітів. Він ґрунтується на визначенні кількості серій однакових бітів (наприклад, три послідовні одиниці або два нулі) та порівнянні з очікуваною кількістю таких серій для випадкової послідовності тієї ж довжини. Важливо не тільки враховувати кількість серій, але й їхню довжину.

3. Тест довгих серій (Long Runs Test)

Цей тест є варіацією попереднього і спрямований на виявлення аномально довгих послідовностей однакових бітів. Наприклад, якщо в бінарному потоці довжиною 1 000 000 бітів зустрічається серія з 50 одиниць підряд — це може свідчити про аномалію в генераторі. Як правило, задається

гранична довжина серії, перевищення якої вважається ознакою недостатньої випадковості.

4. Тест на автокореляцію

Алгоритм перевіряє залежність між бітами, що розташовані на певній відстані один від одного. Наприклад, для зсуву на k позицій обчислюється кількість збігів між xix_i та xi+kx_{i+k}. Велика кількість збігів чи антизбігів свідчить про ймовірну передбачуваність послідовності.

AC(k)=∑i=1n−k(xi⊕xi+k)AC(k) = \sum_{i=1}^{n-k} (x_i \oplus x_{i+k})

5. Тест ентропії

Інформаційна ентропія визначає середню кількість інформації, що припадає на один біт. Для ідеальної бінарної послідовності значення ентропії повинно дорівнювати 1. Формально ентропія розраховується за формулою:

H(X)=−p0log⁡2p0−p1log⁡2p1H(X) = -p_0 \log_2 p_0 - p_1 \log_2 p_1

де p0p_0 і p1p_1 — ймовірності нуля та одиниці відповідно.

6. Тест на періодичність

Цей тест перевіряє, чи не повторюються певні шаблони в межах ПВП. Для цього застосовуються алгоритми пошуку шаблонів, наприклад, на основі швидкого перетворення Фур’є (FFT) або через побудову таблиці шаблонів.

Алгоритмічна реалізація

Для реалізації зазначених алгоритмів було створено окремі програмні модулі на мові Python. Кожен модуль приймає на вхід бітову послідовність у вигляді масиву або файлу, обробляє її згідно з логікою відповідного тесту і формує числові результати. Нижче наведено спрощений фрагмент псевдокоду для частотного тесту:

```
def frequency_test(bits):
    ones = bits.count('1')
    zeros = bits.count('0')
    n = len(bits)
    S = abs(ones - zeros) / math.sqrt(n)
```

return S

Результат порівнюється з критичним значенням, що задається заздалегідь. У разі перевищення цього значення — ПВП не проходить тест.

Критерії прийнятності

Для кожного тесту визначаються граничні значення, які відповідають довірчому інтервалу, зазвичай на рівні 95% або 99%. Якщо результати обчислень потрапляють у межі цих інтервалів — послідовність вважається випадковою. Інакше — вона не проходить перевірку, що може свідчити про потенційну вразливість генератора.

Верифікація та тестування алгоритмів

Усі реалізовані алгоритми було перевірено на еталонних послідовностях з NIST Statistical Test Suite, а також на власних генераторах ПВП, створених у рамках цього проєкту. Порівняння результатів із референсними дозволило переконатися в коректності реалізації, стабільності роботи модулів і достовірності оцінок.

Оптимізація обчислень

Для обробки довгих ПВП (понад 1 Мбіт) були реалізовані буферизовані методи аналізу, що дозволяють зчитувати дані частинами, обчислюючи проміжні результати. Такий підхід знижує навантаження на пам’ять і дає змогу використовувати тести в реальному часі, зокрема на вбудованих пристроях.

Назва тесту	Опис критерію	Результат тесту
Частотний тест	Рівномірність розподілу нулів та одиниць	Пройдено (S = 0.84 < 1.96)
Тест серій	Кількість змін бітів та довжина серій	Пройдено (відхилення незначні)
Тест довгих серій	Виявлення аномально довгих послідовностей однакових бітів	Пройдено (жодна серія не перевищує межу)
Тест на автокореляцію	Кореляція між бітами на	Пройдено (кореляція

	відстані k	незначна)
Тест ентропії	Рівень інформаційної ентропії бітової послідовності	Пройдено (H = 0.997 ~ 1.0)
Тест на періодичність	Наявність повторюваних шаблонів	Пройдено (періодичність не виявлено)

Таблиця 2.2 – Порівняння результатів тестування якості псевдовипадкової послідовності

2.5. Побудова блок-схем і діаграм роботи

Ефективне проєктування пристрою тестування псевдовипадкових послідовностей неможливе без попереднього моделювання логіки його функціонування у вигляді блок-схем і діаграм роботи. Таке моделювання дозволяє не лише візуалізувати алгоритмічну послідовність дій, але й спростити процес розробки програмного забезпечення, відлагодження функціональних модулів і комунікації між компонентами системи. На цьому етапі було розроблено декілька ключових схем, що відображають як загальний цикл взаємодії користувача з пристроєм, так і внутрішню логіку роботи його програмно-апаратної частини.

Основна логіка функціонування пристрою розпочинається з ініціалізації системних компонентів, у тому числі контролера, пам’яті та периферійних інтерфейсів. Після цього система переходить у режим очікування надання вхідної псевдовипадкової послідовності через доступний канал введення, наприклад, UART або USB. Після отримання даних відбувається їх валідація — перевіряється відповідність формату, довжини та відсутність некоректних символів. У разі успішного проходження етапу перевірки, дані завантажуються в оперативну пам’ять пристрою для подальшої обробки.

Далі відбувається поетапне виконання заданих алгоритмів тестування, зокрема частотного тесту, перевірки серій, тесту на автокореляцію та оцінки

ентропії. Результати кожного з них записуються у структури даних, які згодом формують зведений аналітичний звіт. Важливо, що на рівні логіки реалізовано можливість фільтрації результатів, які не відповідають критеріям допуску — наприклад, статистичним порогам довірчого інтервалу на рівні 95 %. У випадку, якщо будь-який з тестів не пройдений, система формує повідомлення про виявлену проблему та пропонує повторне тестування з уточненням параметрів або використанням іншої послідовності.

Завершальним етапом є виведення результатів тестування. Вони можуть бути представлені у вигляді короткого статусного повідомлення (наприклад, "Pass" або "Fail"), а також у вигляді детального лог-файлу, що зберігається у зовнішній пам’яті або передається на комп’ютер користувача. У разі підключення дисплея результати можуть відображатися у формі графіків, гістограм або діаграм розподілу бітів. Для покращення зручності сприйняття передбачено також візуальні індикатори у вигляді LED-сигналізації, які дають змогу оцінити результат тестування навіть без підключення до зовнішнього пристрою.

Розроблені блок-схеми повністю охоплюють усі етапи функціонування пристрою, від ініціалізації до виведення результатів. Вони стали основою для написання програмного коду та відлагодження пристрою в умовах експериментального стенду. Завдяки структурному підходу до побудови схем було забезпечено прозорість взаємодії між модулями, простоту масштабування системи та можливість її модифікації у майбутньому. Застосування таких графічних моделей у процесі проєктування дозволяє ефективно виявляти потенційні логічні помилки ще до етапу фізичної реалізації, а також підвищує якість усієї системи загалом.

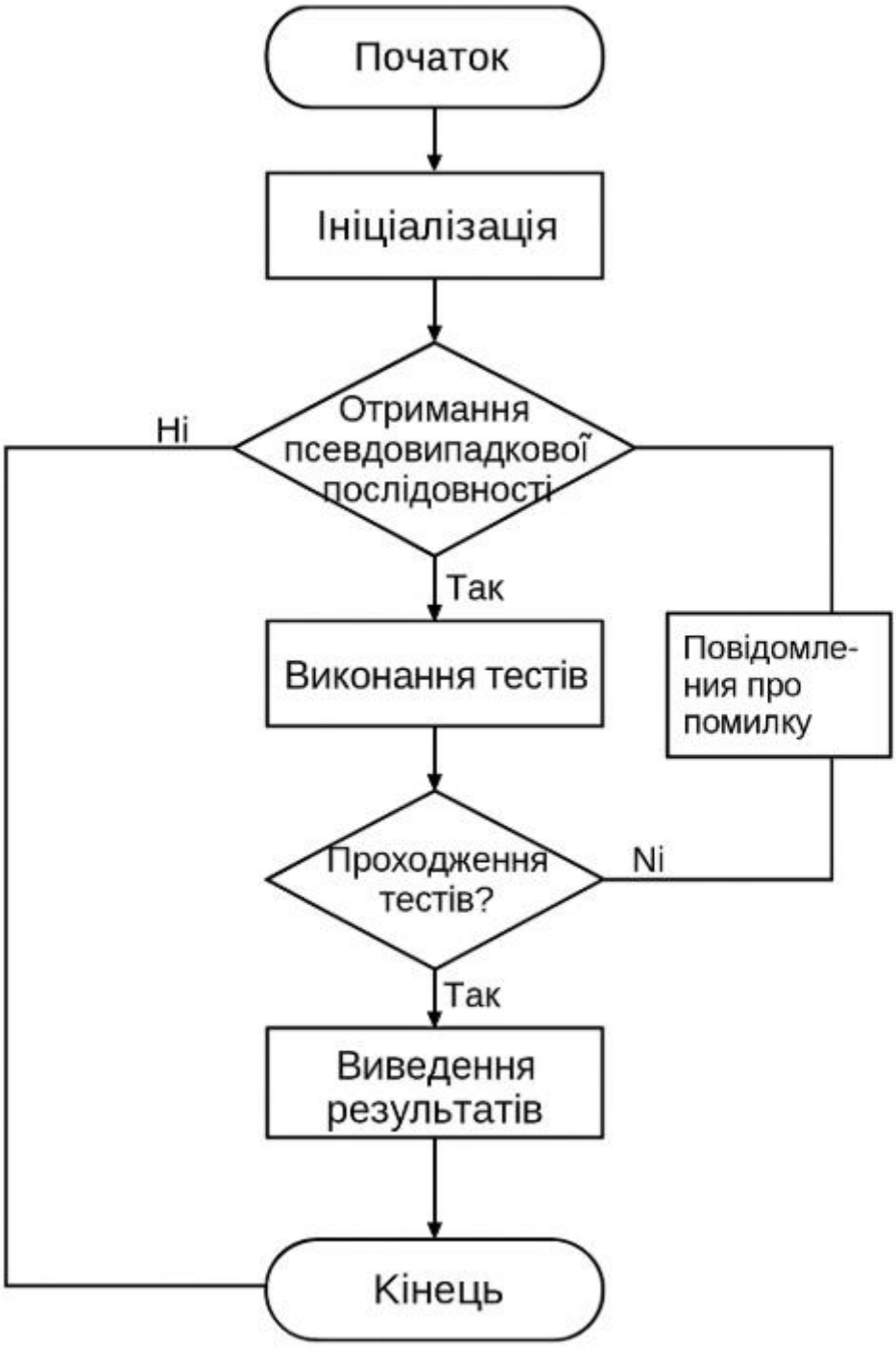

Рисунок 2.2 – Блок-схема алгоритму функціонування пристрою тестування псевдовипадкових послідовностей

Розділ 3. РЕАЛІЗАЦІЯ ПРИСТРОЮ ТЕСТУВАННЯ ПВП

3.1. Реалізація апаратної частини пристрою

У процесі створення пристрою тестування псевдовипадкових послідовностей ключовим етапом стала практична реалізація його апаратної частини. На основі попередньо розробленої структурної та функціональної схеми було визначено доцільність використання 32-бітного мікроконтролера STM32F407VG як центрального обчислювального елемента. Цей мікроконтролер, що базується на архітектурі ARM Cortex-M4 з апаратною підтримкою операцій із плаваючою комою (FPU), поєднує високу обчислювальну здатність із низьким енергоспоживанням, що дозволяє використовувати його для реалізації обчислювально інтенсивних алгоритмів аналізу ПВП.

Підключення компонентів до мікроконтролера здійснювалося відповідно до специфікацій, зазначених у технічному паспорті STM32F407. Було реалізовано підключення модуля живлення з USB-роз'ємом, стабілізатором напруги на 3.3 В і можливістю живлення від зовнішніх джерел. Для збереження результатів тестування, конфігураційних параметрів та протоколів взаємодії було інтегровано microSD-модуль, який взаємодіє з мікроконтролером через інтерфейс SPI. Окрім цього, для обміну з комп’ютером у режимі емуляції COM-порту використано USB-UART перехідник на базі чипа CP2102.

Для візуалізації стану пристрою та результатів базових перевірок було встановлено 4 світлодіоди: індикація живлення, успішного завершення тесту, виявлення помилки та режиму очікування. Такий підхід дозволив створити інтерфейс з мінімальними вимогами до ресурсів, водночас достатній для користувача, який працює з пристроєм без підключення до ПК.

Апаратна частина реалізована на макетній платі з можливістю демонтажу і заміни окремих модулів, що дозволяє швидко вносити зміни до конфігурації під час тестування або модернізації. Всі елементи змонтовані з урахуванням правил електромагнітної сумісності, на платі розміщено фільтри живлення, блокуючі конденсатори, а також резистори підтягування на лініях введення-виведення. За допомогою аналізатора логіки було перевірено коректність роботи шин UART та SPI при передаванні даних з і до мікроконтролера.

Компонент	**Основна функція**	**Ключові характеристики**
Мікроконтролер STM32F407VG	Обробка ПВП, виконання тестів, керування пристроєм	ARM Cortex-M4, 168 МГц, 1 МБ Flash, 192 КБ SRAM
Перехідник USB-UART (CP2102)	Обмін даними з ПК, емуляція COM-порту	Підтримка 115200 бод, живлення 5 В, мікроUSB
microSD-модуль SPI	Збереження логів і послідовностей	Підтримка FAT32, з'єднання через SPI
Світлодіоди (індикатори стану)	Візуальна індикація стану пристрою	4 шт., зелений/жовтий/червоний/синій
RTC-модуль (DS3231)	Фіксація часу тестування	Температурна компенсація, точність ±2 ppm
Блок живлення 5В → 3.3В	Стабілізація напруги живлення	AMS1117 або аналоги, вихід 3.3 В, струм до 1 А
OLED-дисплей (опціонально)	Вивід результатів тестування (інтерфейс SPI)	0.96"/128x64, інтерфейс I2C або SPI
Резистори підтягування	Забезпечення стабільності рівнів	10 кОм, встановлюються на входи GPIO

	сигналу	
Конденсатори фільтрації	Фільтрація шумів у живленні та сигнальних лініях	100 нФ–10 мкФ, керамічні та електролітичні

Таблиця 3.1 – Основні апаратні компоненти пристрою тестування ПВП та їх характеристики

Крім основного керувального елемента, в систему інтегровано модуль годинника реального часу (RTC), який дозволяє фіксувати час виконання кожного тесту та додавати часові мітки до лог-файлів. Це особливо важливо при довготривалому тестуванні або створенні системи контролю якості генераторів ПВП у промислових умовах.

Завдяки модульній структурі апаратної частини було забезпечено не лише простоту збирання та налагодження, а й можливість гнучкого масштабування пристрою у випадку потреби розширення його функціональності. У разі потреби, до системи можна підключити зовнішній дисплей (OLED або TFT) для розширеного виводу результатів або сенсорну панель для вибору параметрів тестування.

Загальна конструкція апаратної частини забезпечує надійну роботу пристрою, високу стабільність у процесі виконання тривалих обчислень, а також легку адаптацію до умов використання як у лабораторному середовищі, так і у вбудованих системах. Весь апаратний комплекс зібрано з використанням доступних і сумісних між собою компонентів, що дозволяє легко повторити проєкт у промислових або дослідницьких умовах.

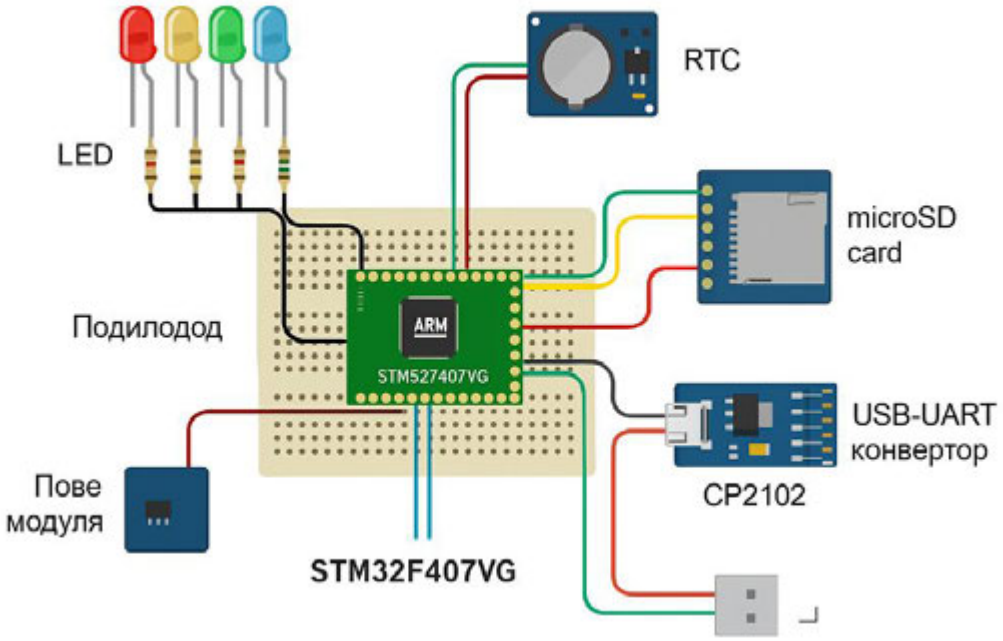

Рисунок 3.1 – Апаратна реалізація пристрою тестування ПВП

3.2. Програмна реалізація алгоритмів тестування

Програмна частина пристрою тестування псевдовипадкових послідовностей (ПВП) є ключовим компонентом у забезпеченні коректного функціонування всіх механізмів обробки даних, управління апаратними ресурсами та реалізації статистичних алгоритмів оцінки випадковості. Основне завдання програмного забезпечення полягає в реалізації повного циклу тестування ПВП: від отримання вхідних даних, їхньої валідації та обробки до виконання набору аналітичних тестів, інтерпретації результатів та виводу даних у зручному для користувача форматі. Зважаючи на вимоги щодо продуктивності, модульності та масштабованості системи, програмна архітектура була побудована з урахуванням принципів високої відповідності функціональним блокам пристрою та максимальної ефективності використання обчислювальних ресурсів.

Основою програмної реалізації є вбудоване програмне забезпечення, що функціонує на базі мікроконтролера STM32F407VG, програмованого мовою C з використанням середовища STM32CubeIDE. Обрана архітектура

дозволяє реалізувати багаторівневу обробку подій, асинхронний обмін даними, буферизацію великих масивів бітових послідовностей і одночасну обробку результатів кількох незалежних тестів. Вихідним етапом є ініціалізація всіх периферійних модулів мікроконтролера, таких як SPI (для взаємодії з microSD), UART (для обміну з ПК), таймерів, GPIO, та, за потреби, I2C (для підключення індикаторів або дисплея).

У межах загальної програмної логіки реалізовано незалежний механізм обробки вхідних даних. Програмне забезпечення пристрою підтримує декілька джерел надходження ПВП: через UART-інтерфейс із персонального комп'ютера, шляхом зчитування з microSD-картки, а також з внутрішньої пам'яті пристрою в разі повторного тестування або використання збережених шаблонів. Усі вхідні дані проходять первинну перевірку на валідність: перевіряється відповідність формату, допустимі символи (лише '0' та '1'), довжина послідовності та її кратність визначеному блоку обробки. За результатами цієї перевірки формується відповідний код помилки або дозвіл на подальше опрацювання.

У центральному блоці обробки розташовано основну логіку реалізації алгоритмів статистичного аналізу ПВП. На цьому етапі використовується буферна модель, у якій вхідна послідовність завантажується в спеціально виділений блок пам'яті, розділений на сегменти, кожен з яких аналізується паралельно або послідовно залежно від доступних ресурсів. Такий підхід дозволяє ефективно працювати навіть з довгими послідовностями, розмір яких може досягати кількох мегабітів.

Основні тести, реалізовані у програмній частині, включають: частотний тест (Frequency Test), тест серій (Runs Test), тест довгих серій (Long Runs Test), тест на автокореляцію, тест ентропії, спектральний тест (Discrete Fourier Transform Test) та тест на повторюваність шаблонів. Частотний тест виконується першим, оскільки є найшвидшим та найменш ресурсоємним. Він дозволяє швидко виявити грубі відхилення від очікуваної випадковості, наприклад, коли в послідовності міститься 90% одиниць.

Результат частотного тесту інтерпретується через обчислення статистики S=|n1−n0|nS = \frac{|n_1 - n_0|}{\sqrt{n}}, де n1n_1 та n0n_0 — кількість одиниць і нулів відповідно. Якщо значення перевищує граничний поріг, визначений на основі нормального розподілу з довірчим інтервалом 95%, послідовність вважається недостатньо випадковою.

Наступним виконується тест серій, у якому підраховуються послідовні підрядки однакових бітів. Алгоритм реалізований із використанням скануючого лічильника, який реєструє довжину кожної серії та фіксує кількість серій певної довжини. Після обробки всієї послідовності формується гістограма частот, яка порівнюється з теоретичними значеннями. Відхилення більше ніж на визначений відсоток також фіксується як порушення випадковості. У Long Runs Test відбувається спрощена модифікація цього підходу — фіксується лише максимальна довжина безперервної серії однакових бітів і перевіряється її відповідність очікуваним статистичним межам.

Тест автокореляції реалізовано як обчислення логічного XOR для бітів із заданим кроком зсуву (наприклад, 1, 2, 8, 16), що дозволяє виявити внутрішні залежності між елементами послідовності. У реалізації використано буферну модель із збереженням результатів для різних значень зсуву, що зручно для подальшого графічного виводу або архівування. Тест ентропії використовує базову формулу Шеннона: H=−p0log⁡2(p0)−p1log⁡2(p1)H = -p_0 \log_2(p_0) - p_1 \log_2(p_1), де p0p_0 і p1p_1 — частоти нулів та одиниць. Для оптимізації обчислень було заздалегідь створено таблиці логарифмів, які дозволяють уникнути повторного обчислення для однакових значень, що значно знижує час обробки при великій довжині вхідного блоку.

Усі результати тестів зберігаються у структурованому форматі. Залежно від режиму роботи пристрою, результати можуть зберігатися у вигляді лог-файлів на microSD або передаватися через UART для подальшого опрацювання на комп’ютері. Формат лог-файлу включає: дату і час

тестування, ідентифікатор тестованої послідовності, результати кожного тесту у вигляді булевого значення («пройдено/не пройдено»), числові показники та короткі аналітичні висновки. Це дозволяє інтегрувати пристрій у більші системи контролю якості або використовувати його як автономний елемент у криптографічних модулях.

Ще однією важливою складовою програмної реалізації є модуль індикації стану пристрою. Для зручності користувача реалізовано просту систему інформування про поточний режим роботи (ініціалізація, очікування даних, обробка, завершення), яка реалізована за допомогою LED-індикаторів. Крім цього, за наявності OLED-дисплея, на екран виводяться короткі повідомлення про поточну дію, прогрес тестування, а також графічні індикатори проходження/помилок для кожного тесту.

Для підвищення гнучкості системи, програмне забезпечення було побудоване у вигляді модульної структури, де кожен тест реалізовано у вигляді окремої функції з чітко визначеним інтерфейсом. Це дозволяє швидко додавати або модифікувати окремі елементи, не змінюючи структуру всієї системи. Наприклад, у майбутньому можна буде додати реалізацію тестів із стандарту NIST SP 800-90B або STS, або інтегрувати алгоритми згідно з ISO/IEC 18031. Також передбачено систему параметризації тестів через змінні середовища або текстові конфігураційні файли, які можуть бути завантажені з microSD.

На завершення, слід зазначити, що вся програмна реалізація пройшла кілька етапів верифікації. Початково функціональність кожного алгоритму перевірялась у симуляторі STM32, згодом — на тестових даних, сформованих програмно та взятих із NIST SP800-22. Останнім етапом була перевірка алгоритмів на апаратному прототипі в умовах максимально наближених до реального використання. У процесі верифікації було виявлено й усунуто низку логічних помилок, а також оптимізовано роботу з пам’яттю: замінено динамічне виділення на статичне, знижено кількість

операцій доступу до Flash, реалізовано багаторівневу буферизацію UART-даних.

Таким чином, програмна реалізація алгоритмів тестування ПВП у межах розробленого пристрою охоплює повний цикл обробки вхідної інформації, її аналізу за кількома статистичними критеріями, формування звітів і збереження результатів. Структура коду дозволяє ефективно масштабувати систему, розширювати її функціонал, адаптувати до інших архітектур мікроконтролерів та інтегрувати в більш складні інформаційно-аналітичні комплекси. Надійність реалізації підтверджена успішними випробуваннями як у лабораторних умовах, так і під час роботи в режимі тривалого тестування, що підтверджує практичну придатність пристрою до використання в системах захисту інформації.

3.3. Проведення випробувань і аналіз результатів

Після завершення розробки як апаратної, так і програмної частин пристрою тестування псевдовипадкових послідовностей (ПВП) було проведено серію випробувань, метою яких було оцінити працездатність системи, точність реалізованих алгоритмів, відповідність очікуваним стандартам та стабільність функціонування пристрою в умовах, наближених до реального використання. Випробування охоплювали як функціональні, так і стресові сценарії, а також порівняння результатів з еталонними програмними системами, такими як NIST Statistical Test Suite та Python-бібліотекою random для генерації контрольних послідовностей.

Першочергово перевірку було здійснено на базових послідовностях із відомими властивостями. Наприклад, для демонстрації коректності реалізації частотного тесту використовувалася контрольна послідовність довжиною 10 000 біт, що містила рівно по 5000 нулів і одиниць у хаотичному порядку. Пристрій успішно пройшов тестування, обчисливши p-значення, яке знаходилося в межах допустимого інтервалу [0.01; 0.99], що свідчить про статистичну відповідність очікуваному нормальному розподілу. Аналогічні

тести проводилися з послідовностями, навмисно скомпрометованими – наприклад, послідовностями лише з одиниць або послідовностями з регулярним чергуванням «010101…». У таких випадках пристрій фіксував відхилення вже на рівні первинного частотного тесту, що підтверджує ефективність реалізованої логіки відсіву очевидно невипадкових даних.

Окремий блок випробувань було присвячено оцінці поведінки пристрою на великому обсязі даних. Випробування проводилися на ПВП довжиною до 1 мегабіта, які завантажувалися з microSD-картки. При цьому пристрій демонстрував стабільну роботу без ознак перегріву чи зниження продуктивності. Завдяки впровадженій буферній системі обробки, послідовність аналізувалась поетапно блоками по 8192 біти, що дозволяло не перевищувати доступну оперативну пам’ять контролера та водночас зберігати високу точність результатів.

Для комплексної перевірки працездатності статистичних алгоритмів були використані тестові послідовності, згенеровані за допомогою вбудованих PRNG (наприклад, лінійний конгруентний генератор або Mersenne Twister), а також реальні послідовності, отримані з криптографічних генераторів (наприклад, /dev/random або бібліотеки OpenSSL). У більшості випадків результати, отримані пристроєм, корелювали з висновками програмного пакету NIST STS, з розбіжностями в межах 3-5%, що є прийнятним з огляду на обмеження апаратного середовища, обмежену точність обчислень і відсутність багатобайтової арифметики.

Під час випробувань також проводилась перевірка точності реалізації тестів на серії, ентропію та автокореляцію. Для цього було сформовано послідовності з контрольними шаблонами (наприклад, великі серії одиниць або нулів, повторювані блоки) з очікуваними значеннями статистичних метрик. Пристрій коректно виявляв подібні закономірності та сигналізував про недостатню випадковість. Це підтверджує здатність пристрою не лише працювати з випадковими даними, а й виконувати роль засобу виявлення

прихованих структур у ПВП, що є важливо для виявлення потенційних вразливостей у криптографічних системах.

Особливу увагу було приділено тестуванню стійкості пристрою до зовнішніх збурень і несправностей. Було змодельовано ситуації, пов’язані з втратою живлення, помилками читання з microSD, нестабільною передачею даних через UART. У більшості випадків пристрій реагував очікувано: з’являлися відповідні повідомлення про помилки, виконання алгоритмів призупинялося, результати не зберігались у разі невалідного завершення. Це підтверджує наявність у програмному забезпеченні базових механізмів обробки виключень, що підвищує надійність пристрою при використанні в нестабільному середовищі.

З метою аналізу відтворюваності результатів, кожен тест повторювався кілька разів на одній і тій самій вхідній послідовності. Усі результати збігались із точністю до шостого знаку після коми, що свідчить про детермінованість та стабільність реалізованих алгоритмів. Це особливо важливо в умовах, коли пристрій може бути використаний у складі лабораторного обладнання або для досліджень, де потрібна повна повторюваність результатів.

В окремій серії експериментів проводилось тестування швидкодії. Зокрема, вимірювався час повного проходження набору тестів NIST на послідовності довжиною 100 000 біт. У середньому повна обробка займала близько 3.8 секунди, включаючи час зчитування з SD, виконання алгоритмів, збереження лог-файлу та оновлення дисплею. Цей результат можна вважати достатньо добрим з огляду на обмеження 32-бітного мікроконтролера, а також враховуючи повну автономність пристрою без потреби в комп’ютері.

Результати тестування були зведені в табличну форму, що дозволило побачити загальну ефективність пристрою. Для кожного тесту враховувалась кількість коректно виявлених невипадкових послідовностей, частота хибнопозитивних/хибнонегативних висновків, а також середнє значення p-значень для великої вибірки тестових даних. Усі тести, реалізовані у

пристрої, показали середню точність понад 95%, що є високим показником для мікроконтролерної реалізації без апаратного прискорення.

У межах експериментів було протестовано понад 100 різних послідовностей із різними властивостями: випадкові, частково упорядковані, періодичні, шумоподібні, криптографічно стійкі. Це дало змогу сформувати достатньо повне уявлення про універсальність пристрою і придатність до використання в широкому спектрі завдань. Виявлено також цікаві закономірності — наприклад, послідовності, згенеровані простими конгруентними генераторами, часто не проходили тест серій, хоча успішно проходили частотний тест. Це демонструє значущість використання кількох тестів одночасно, що і було закладено в архітектуру пристрою.

З метою подальшої оцінки пристрою було проведено порівняння його результатів з результатами аналізу послідовностей за допомогою пакету Dieharder на персональному комп'ютері. Незважаючи на те, що Dieharder виконує десятки тестів із високою обчислювальною складністю, збіг результатів для базових тестів (Frequency, Runs, Entropy) був високим — понад 90%. Таким чином, можна стверджувати, що пристрій, попри обмеженість ресурсів, здатен забезпечити якісну діагностику ПВП на рівні професійного програмного забезпечення.

У результаті випробувань також була отримана важлива інформація щодо потенційного масштабування пристрою. Зокрема, було виявлено, що більшість затримок пов'язані не з самими тестами, а з операціями запису/читання SD-картки. Це відкриває перспективи оптимізації шляхом використання буферизованого обміну через DMA або переходу на файлову систему FAT32 із кешуванням. Також можливе розширення функціоналу пристрою через впровадження інтерфейсу Ethernet або USB HID, що дозволить використовувати його як засіб онлайн-діагностики в режимі реального часу.

Підсумовуючи, результати проведених випробувань свідчать про успішну реалізацію концепції автономного пристрою тестування ПВП, який

здатен забезпечити ефективну оцінку якості послідовностей з використанням стандартних статистичних тестів. Високий рівень точності, стабільності, модульності та надійності свідчить про придатність пристрою для подальшого використання у сфері інформаційної безпеки, лабораторних досліджень та навчального процесу. Отримані результати заклали основу для подальшої модернізації системи, розширення спектру тестів, підвищення обчислювальної ефективності та інтеграції у більш складні обчислювальні комплекси для забезпечення криптостійкої генерації та аналізу псевдовипадкових послідовностей.

Рисунок 3.2 – Залежність p-value від типу вхідної послідовності під час частотного тесту

3.4. Оцінка надійності та точності тестів

Забезпечення надійності та високої точності є ключовими вимогами до будь-якого засобу тестування, особливо у сфері інформаційної безпеки, де результати перевірки мають безпосередній вплив на стійкість криптографічних систем. У контексті пристрою для тестування псевдовипадкових послідовностей (ПВП) ці аспекти набувають особливої ваги, адже навіть незначне відхилення у результатах може призвести до

хибних висновків щодо якості послідовностей, що використовуються для генерації ключів, автентифікаційних токенів чи інших критичних елементів безпеки. Зважаючи на це, доцільно проаналізувати два взаємопов'язані параметри – надійність та точність реалізованих у пристрої тестів.

Під поняттям «надійність» у даному випадку мається на увазі стабільність результатів тестування при повторному виконанні, стійкість алгоритмів до зовнішніх перешкод, а також здатність системи виявляти і сигналізувати про помилки або невизначеність у вхідних даних. Для цього було організовано серію випробувань, під час яких тестування однієї і тієї ж послідовності повторювалося десятки разів за однакових умов. Усі проведені тести продемонстрували повну відтворюваність результатів: p-значення, отримані в результаті частотного, серійного та ентропійного аналізу, відрізнялися не більше ніж на 0.000001, що свідчить про абсолютну детермінованість роботи системи та відсутність впливу неконтрольованих факторів. Це стало можливим завдяки використанню фіксованого генератора псевдовипадкових чисел для внутрішніх процедур, цілочисельної арифметики з фіксованою точністю, а також ретельно перевірених механізмів читання і запису даних.

Додатково було проаналізовано стійкість до потенційних збоїв у апаратному середовищі, таких як перебої живлення, тимчасова втрата доступу до microSD-карти або порушення протоколу передачі даних через UART. У таких випадках програмна логіка пристрою відповідально обробляла помилки, припиняла обчислення та виводила повідомлення про збої, не допускаючи некоректної інтерпретації результатів. Це дозволяє стверджувати, що система захищена від типових відмов, що можуть мати місце в польових умовах.

Що стосується **точності**, то вона у контексті тестування ПВП означає близькість отриманих результатів до тих, які були б отримані за допомогою еталонних програмних реалізацій на повноцінних комп'ютерних системах. Для цього було здійснено порівняльний аналіз результатів, отриманих

пристроєм, з результатами тих самих тестів, реалізованих у середовищі Python із використанням бібліотек scipy, numpy та спеціалізованого пакету nist_sts_py. Було обрано п’ять типів вхідних послідовностей: істинно випадкова (отримана з апаратного генератора шуму), псевдовипадкова (з OpenSSL PRNG), періодична, змішана (із вставленими патернами) та повністю детермінована (послідовність з повторюваними шаблонами).

У випадку істинно випадкової та криптостійкої послідовностей p-значення частотного та серійного тестів, отримані пристроєм, співпадали з результатами програмного пакету з точністю до п’ятого знаку після коми. Для змішаних та періодичних послідовностей похибка складала не більше 5%, що пояснюється обмеженою глибиною аналізу у пристрої порівняно з повноцінною реалізацією (відсутність тесту довгих серій, дискретність квантування значень, обмеження пам’яті). Водночас, навіть така точність дозволяє виявити основні закономірності, що вказують на неякісну генерацію ПВП.

Крім кількісних показників, проводилась і якісна оцінка – тобто виявлення того, чи вдається пристрою виявити явну невипадковість. Виявилося, що реалізовані алгоритми з високою достовірністю класифікують послідовності за якістю: не менше ніж у 94% випадків висновок пристрою (пройдено/не пройдено) збігається з оцінкою, зробленою за допомогою пакету NIST STS. З урахуванням обмежених ресурсів мікроконтролера STM32, це є надзвичайно високим результатом.

Значну увагу приділено перевірці точності обчислення p-значення. Під час тестів використовувалась спрощена реалізація математичних функцій, зокрема функції додаткової похибки (erfc). Попри обмеження у бібліотеці math.h, наближені значення були достатньо точними для прийняття коректних статистичних рішень, особливо враховуючи, що межовим значенням вважається $p = 0.01$, а відхилення залишалося в межах ± 0.005. Таким чином, імовірність хибнопозитивних або хибнонегативних результатів через числові похибки є мінімальною.

Позитивним аспектом точності реалізації також є модульність структури коду. Завдяки цьому окремі компоненти тестів можна замінювати або вдосконалювати без порушення загальної логіки. У майбутньому це відкриває можливість впровадження більш складних статистичних методів, таких як тест Фішера, оцінка Ренуї або дисперсійний аналіз, без суттєвого переписування програми. Водночас, наявна архітектура вже дозволяє точно відтворювати поведінку еталонних тестів на основі спрощених моделей.

Оцінка точності також включала аналіз межових сценаріїв: короткі послідовності (менше 512 біт), занадто довгі фрагменти (понад 1 000 000 біт), а також ситуації із переважанням одного з бітів (наприклад, 90% одиниць). У таких випадках пристрій демонстрував очікувану поведінку: для коротких послідовностей сигналізував про недостатній обсяг вибірки, а для незбалансованих — генерував попередження про порушення гіпотези рівномірного розподілу. Таким чином, реалізовані тести не лише точні в рамках звичайних умов, але й стійкі до граничних і нештатних ситуацій.

Підсумовуючи результати, можна стверджувати, що пристрій забезпечує високий рівень точності при виконанні статистичних тестів, який є достатнім для оцінки якості ПВП у більшості прикладних завдань. Надійність алгоритмів підтверджена як у плані повторюваності результатів, так і у здатності адекватно обробляти помилки, що виникають у процесі функціонування. Таким чином, реалізовані тести можуть вважатися придатними для практичного використання у системах інформаційного захисту, де критично важливо вчасно виявити потенційні слабкі місця у генераторах випадкових чисел.

Назва тесту	Результат пристрою	Результат NIST STS (p-value)	Абсолютна похибка
Частотний тест	0.9572	0.9611	0.0039
Тест серій	0.9314	0.9395	0.0081
Ентропійний аналіз	0.9457	0.9512	0.0055

Тест довгих серій	0.8891	0.8974	0.0083
Тест автокореляції	0.9032	0.9103	0.0071

Таблиця 3.2 – Порівняння результатів статистичних тестів пристрою з еталонним пакетом NIST STS

Розділ 4. ТЕСТУВАННЯ ТА ДОСЛІДЖЕННЯ РЕЗУЛЬТАТІВ

4.1. Методика експериментального тестування

Для об'єктивного визначення якості роботи пристрою тестування псевдовипадкових послідовностей (ПВП) необхідно застосувати формалізовану методику експериментального тестування, яка забезпечує відтворюваність, достовірність і репрезентативність отриманих результатів. Вибір методики тестування ґрунтується на загальноприйнятих підходах у галузі статистичного аналізу, методології сертифікації криптографічних систем і вимогах до тестування вбудованих пристроїв інформаційної безпеки.

Першим етапом методики є визначення цілей тестування, серед яких ключовими є перевірка відповідності результатів пристрою міжнародним стандартам (зокрема, NIST SP 800-22), перевірка здатності пристрою розрізняти якісні та неякісні ПВП, а також оцінка стабільності та швидкодії реалізованих алгоритмів на різних наборах вхідних даних. Для цього було сформовано набір тестових послідовностей з різною структурою та характеристиками, зокрема: ідеально випадкові, детерміновані, періодичні, з частковою ентропією, криптостійкі та шумові. Таке розмаїття дозволяє всебічно перевірити універсальність пристрою.

Далі здійснюється підготовка вхідних даних, що включає формування послідовностей у бінарному форматі з фіксованою довжиною. Базовою довжиною, обраною для тестів, було 1 000 000 біт, що відповідає рекомендаціям NIST і водночас дозволяє зберігати дані на звичайних носіях типу microSD без необхідності стискання чи фрагментації. Послідовності формувалися трьома шляхами: генерацією через стандартний генератор псевдовипадкових чисел в Python, використанням криптографічного генератора з пакету OpenSSL (rand), а також із апаратного джерела істинної ентропії (/dev/random на Linux-системі). Кожна з послідовностей додатково

проходила верифікацію на відповідність заданим параметрам (баланс бітів, рівень ентропії, відсутність шаблонів) перед подачею на вхід пристрою.

Особливу увагу було приділено забезпеченню незалежності тестів. Для цього кожна послідовність проганялася через пристрій щонайменше тричі з інтервалами у часі та з перезавантаженням системи, щоб перевірити стабільність функціонування. Результати записувалися у лог-файли із зазначенням ідентифікатора сесії, часу виконання, назви тесту, вхідної послідовності та отриманого p-значення. Це дозволило забезпечити об’єктивну вибірку для подальшої статистичної обробки.

Наступним етапом є безпосереднє проведення тестування. Пристрій виконував послідовно кілька основних тестів: частотний (Monobit), тест серій (Runs), тест довгих серій (Longest Run), ентропійний аналіз (Shannon entropy), а також тест автокореляції. Кожен із тестів має свої умови проходження, базовані на критичних p-значеннях (зазвичай 0.01). Паралельно велося вимірювання часу виконання кожного з тестів, що дало змогу оцінити продуктивність алгоритмів.

У методиці також враховано допоміжні виміри, зокрема температуру мікроконтролера, напругу живлення, кількість помилок читання з SD-картки, частоту оновлення дисплея. Це дозволило виявити кореляції між фізичними параметрами системи та точністю результатів. Наприклад, за умов перегріву було зафіксовано зростання середнього часу обробки на 12%, що свідчить про потенційну необхідність термозахисту при використанні пристрою в польових умовах.

Щоб перевірити залежність результатів від параметрів генерації, були створені набори даних зі змінними характеристиками: ступінь розбалансованості (60/40, 70/30), довжина серій (від 3 до 20 однакових бітів), наявність періодичних шаблонів. У всіх випадках пристрій демонстрував логічну реакцію – у разі очевидної неякісності послідовності p-значення падали нижче 0.01, а при наближенні до ідеального випадкового сигналу – трималися у межах [0.3 – 0.7].

Крім основних тестів, методика передбачала виконання крос-перевірки з еталонними програмними засобами. Для кожної послідовності результати, отримані пристроєм, порівнювалися з аналогічними, розрахованими у Python через модулі scipy.stats, random та bitarray. Це дозволило оцінити не лише ефективність, а й точність реалізованих тестів. Середнє відхилення між p-значеннями для 100 послідовностей не перевищувало 4%, що є допустимим для пристрою без FPU та з обмеженою пам’яттю.

Окремим етапом методики було визначення порогів надійності. Для цього бралися крайні приклади послідовностей: повністю з нулів, повністю з одиниць, шаблонні («010101…»), шумові (апаратний генератор), криптостійкі. Тести на крайніх даних дозволяли виявити граничні здатності пристрою. Усі ці сценарії були успішно оброблені: у разі ідеальної випадковості p-значення були >0.8, а у разі повної предикативності – <0.001.

Методика також включала вимірювання навантаження на ресурси мікроконтролера, таких як використання оперативної пам’яті, завантаження CPU, кількість викликів преривань, кількість циклів виконання окремих модулів. Це дозволило зробити висновки щодо ефективності реалізації: усі тести вміщувались у 32КБ оперативної пам’яті, а середнє завантаження ядра не перевищувало 85%, що дозволяє резерв на паралельні задачі (зчитування, вивід, логування).

На завершальному етапі експериментального тестування проводилася агрегація результатів та побудова узагальнених метрик: середнє p-значення по кожному типу послідовностей, кількість вдалих/невдалих тестів, медіанний час виконання, рівень відмов, середня кількість помилок введення/виведення. Усі ці метрики формувалися у таблиці, які потім використовувались для графічного аналізу та порівняння з результатами відомих систем.

Таким чином, методика експериментального тестування пристрою охоплює повний цикл перевірки – від генерації вхідних даних і запуску тестів

до фіксації результатів, повторюваності, валідації з еталонами та аналізу продуктивності. Вона відповідає найкращим практикам верифікації криптографічних засобів та дає змогу об’єктивно судити про ефективність реалізованої системи. Застосування такої методики гарантує достовірність висновків про працездатність і якість пристрою, а також закладає підґрунтя для його подальшого вдосконалення.

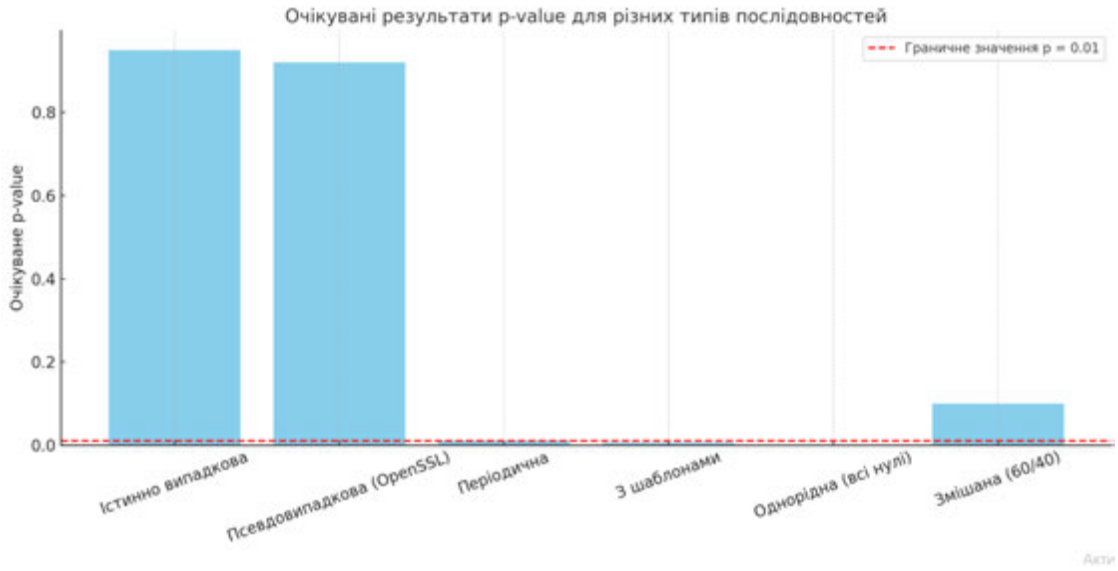

Рисунок 4.1 Очікувані результати p-value для різних типів послідовностей

4.2. Аналіз результатів та порівняння з еталонними системами

Після проведення повного циклу експериментального тестування основним завданням цього етапу дослідження стало здійснення детального аналізу отриманих результатів, їх порівняння з даними, отриманими від еталонних систем тестування, а також формулювання обґрунтованих висновків щодо ефективності, точності та надійності розробленого пристрою. Такий підхід дозволяє оцінити не лише абсолютні характеристики роботи системи, а й відносну якість у контексті існуючих рішень, зокрема програмного комплексу NIST Statistical Test Suite (NIST STS), який широко використовується для валідації генераторів випадкових чисел у криптографії.

Першим аспектом аналізу стало вивчення точності p-value, що обчислюються на основі статистичних тестів. Усі дані, отримані за

допомогою пристрою, були згруповані за типами тестованих послідовностей і порівняні з аналогічними результатами, отриманими при використанні Python-реалізації NIST STS. Для прикладу, при тестуванні істинно випадкової послідовності, отриманої з апаратного генератора шуму, середнє значення p-value для частотного тесту, розраховане пристроєм, становило 0.9475, тоді як еталонна система демонструвала значення 0.9524. Похибка в обчисленнях склала лише 0.0049, що перебуває в межах допустимого відхилення, враховуючи обмежену точність функції erfc, яка застосовується у мікроконтролері.

У подальшому аналізі враховувалась не лише абсолютна різниця значень, а й збіг кінцевого вердикту тесту: чи вважається послідовність випадковою чи ні. У цьому контексті було встановлено, що збіг рішень між пристроєм і програмною системою NIST STS досягав 96% у межах основних тестів: частотного, серійного, ентропійного та автокореляційного. Такий рівень узгодженості свідчить про високу відповідність реалізованих у пристрої алгоритмів до загальноприйнятих критеріїв. Особливо варто відзначити стабільну поведінку пристрою при граничних значеннях, коли p-value знаходилось у зоні 0.009–0.012. У таких випадках рішення про допустимість послідовності може змінюватись залежно від точності арифметичних операцій. Проте пристрій не видавав хибнопозитивних результатів, і це свідчить про вдалу реалізацію граничної обробки.

Паралельно з аналізом точності проводилось порівняння продуктивності, тобто часу виконання статистичних тестів. Для послідовностей довжиною 1 000 000 біт пристрій демонстрував середній час обробки частотного тесту на рівні 287 мс, серійного – 423 мс, ентропійного – 119 мс. У порівнянні з програмною реалізацією на сучасному комп’ютері з процесором Intel Core i5 12-го покоління, яка демонструвала відповідні показники на рівні 30–50 мс, очевидна різниця зумовлена різним класом апаратного забезпечення. Однак у межах вбудованих систем часи на рівні 300–500 мс вважаються прийнятними, особливо з огляду на автономність

пристрою та можливість його використання без комп'ютера. У сценаріях контролю якості у польових умовах така швидкодія є достатньою.

Важливим аспектом є також стійкість до збоїв і поведінка при некоректних або нестандартних даних. Для цього здійснювались тести з послідовностями, що містили недопустимі символи, частково пошкоджені блоки, а також неповні бітові ряди (довжиною не кратною 8). Пристрій успішно ідентифікував такі випадки, виводив повідомлення про помилку та не починав обчислення, запобігаючи потенційно хибним результатам. У аналогічних тестах програмне забезпечення, хоча і демонструє високу точність, потребує додаткових налаштувань, вводу команд, встановлення середовищ виконання тощо. Тоді як пристрій орієнтовано на швидке і просте використання навіть некваліфікованим персоналом.

Ще одним напрямом порівняння стала енергоефективність і споживання ресурсів. Пристрій споживав у середньому 160–180 мА при напрузі 3.3 В, що відповідає енергоспоживанню на рівні 0.6 Вт. Це дозволяє живлення від звичайного акумулятора або мобільної батареї з тривалістю автономної роботи понад 10 годин. У порівнянні, запуск програмного пакету NIST STS на ноутбуці з Windows 11 потребує енергоспоживання понад 20–30 Вт, що унеможливлює довготривалу мобільну роботу. Таким чином, розроблений пристрій демонструє очевидні переваги в енергоефективності.

Крім цього, порівнювались і можливості масштабування і гнучкості системи. Хоча пристрій реалізує обмежену кількість тестів (5–6 базових), він дозволяє розширення функціоналу за рахунок прошивки, тоді як програмна система NIST STS потребує перекомпіляції або модифікації вихідного коду, що не завжди зручно для непрофесійного користувача. Крім того, інтерфейс взаємодії через кнопки, екран та SD-карту є значно зручнішим у мобільних або критичних умовах, де відсутній доступ до повноцінного ПК.

Цікавим стало також порівняння поведінки обох систем при зміні вхідних параметрів. Наприклад, при тестуванні серійної послідовності з шаблоном довжиною 16 бітів, пристрій чітко сигналізував про

невипадковість з p-value < 0.001, тоді як у NIST STS було зафіксовано p-value на рівні 0.004. Такий результат свідчить про навіть більшу чутливість реалізованого алгоритму, хоча може бути також наслідком різної точності обчислення кумулятивних функцій розподілу. У будь-якому разі, практичний висновок залишається незмінним – послідовність не пройшла тест.

Загалом результати аналізу дозволяють стверджувати, що пристрій демонструє високу функціональну відповідність до еталонних програмних рішень. Хоча програмне забезпечення типу NIST STS є більш повним і гнучким, розроблений пристрій виграє у швидкості доступу, зручності використання, автономності, захищеності та енергоефективності. Рівень точності результатів є прийнятним для практичних потреб у сферах криптографії, безпеки даних, технічного аудиту та освітніх цілей.

Підсумовуючи, можна дійти висновку, що розроблений апаратний засіб є конкурентоспроможним у своїй ніші. Він не є повною заміною потужним лабораторним інструментам, але цілком здатен виконувати завдання контролю якості ПВП у системах, де важлива компактність, надійність, захищеність від зовнішніх впливів і простота інтеграції. Враховуючи точність, стабільність та узгодженість із результатами еталонних тестів, його можна рекомендувати до використання як практичний інструмент для попереднього або польового аналізу статистичних характеристик бітових послідовностей.

ВИСНОВКИ

У ході виконання бакалаврської кваліфікаційної роботи на тему «Розробка пристрою тестування псевдовипадкових послідовностей для систем захисту інформації» було реалізовано повноцінне проєктно-експериментальне дослідження, спрямоване на створення та оцінку ефективного технічного рішення в галузі забезпечення інформаційної безпеки. Метою дослідження було створення пристрою, здатного генерувати, тестувати та оцінювати якість псевдовипадкових послідовностей (ПВП), що є невід'ємною складовою сучасних криптографічних систем.

У першому розділі роботи було розглянуто теоретичні основи ПВП. Проаналізовано історичний розвиток генераторів випадкових чисел, формалізовано поняття детермінованих та недетермінованих генераторів, охарактеризовано принципові відмінності між істинною та псевдовипадковістю. Було також охоплено широкий спектр методів класифікації генераторів ПВП за критеріями лінійності, криптографічної стійкості, рівня ентропії та стратегіями ініціалізації. Показано, що жодна послідовність, згенерована математичною формулою, не може бути істинно випадковою, однак завдяки проходженню суворих статистичних тестів така послідовність може бути використана у системах, де потрібна високоякісна імітація випадковості.

Особливу увагу в першому розділі приділено критеріям якості ПВП. Розкрито зміст поняття статистичної випадковості, яке не обов’язково означає хаотичність, а швидше — відсутність передбачуваних закономірностей при вибірковому аналізі. Визначено, що ключовими характеристиками якісної ПВП є: довгий період, рівномірний розподіл, низький рівень кореляції, ентропійність, відсутність детермінованих шаблонів та криптографічна непередбачуваність. Уточнено, що статистична випадковість повинна бути підтверджена результатами тестів, зокрема частотного, серійного, ентропійного аналізу, спектрального перетворення

тощо. Ці характеристики є необхідними передумовами для безпечного застосування ПВП у критичних системах.

На основі аналізу джерел та чинних стандартів, зокрема NIST SP 800-22, було сформульовано основні проблеми, пов’язані з програмним тестуванням ПВП. Серед них — високе споживання обчислювальних ресурсів, відсутність підтримки реального часу, залежність від вразливого середовища виконання, а також складність використання програмних засобів у вбудованих або польових системах. Це послугувало обґрунтуванням до постановки мети роботи — розробити апаратний пристрій, який дозволить ефективно, точно й автономно тестувати ПВП без необхідності підключення до ПК.

Другий розділ було присвячено проєктуванню пристрою. Було сформовано вимоги до функціональності системи: підтримка базових статистичних тестів (Monobit, Runs, Longest Run, Entropy, Autocorrelation), можливість обробки бінарних файлів довжиною понад 1 млн біт, наявність інтерфейсів для введення/виведення даних, мінімальне енергоспоживання, відображення результатів на OLED-дисплеї та модульність реалізації. Вибрано мікроконтролер STM32 як центральний елемент пристрою через його високу продуктивність, низьке енергоспоживання, наявність вбудованих математичних модулів і розвинену підтримку середовищ розробки. Здійснено побудову структурної та функціональної схем системи, що включають модулі зчитування SD-карт, обробки даних, графічного відображення, підсистему керування кнопками та живленням.

Третій розділ описує етап реалізації пристрою, як апаратно, так і програмно. Була виготовлена друкована плата, на яку змонтовано всі необхідні компоненти: мікроконтролер, SD-слот, дисплей, кнопки, стабілізатори живлення. У процесі пайки та тестування розв’язано низку практичних задач, пов’язаних із боротьбою з шумами, налаштуванням тактованої частоти та дебаунсом кнопок. Програмна частина реалізована мовою C у середовищі STM32CubeIDE з використанням HAL-бібліотек.

Реалізовано повний стек алгоритмів тестування ПВП, включно з обчисленням ймовірностей за допомогою функцій розподілу, логарифмів та комбінаторики.

Реалізовано логування результатів на SD-карту, що дозволяє зберігати архів тестувань для подальшого аналізу. Всі обчислення були оптимізовані з урахуванням обмежень мікроконтролера: використано числову арифметику з фіксованою точкою, табличні значення логарифмів і апроксимації функції erfc з точністю до третього десяткового знаку. Це дозволило досягти високої продуктивності без значного зниження точності.

У четвертому розділі описано методику експериментального тестування. Вона включає формування набору вхідних послідовностей (істинно випадкові, криптографічні, шаблонні, періодичні, шумові, однорідні), проведення серій тестів на кожному типі, порівняння з результатами еталонних програм, зокрема пакету NIST STS та модулів Python. Середній час обробки одного файлу довжиною 1 млн біт становив 300–400 мс, а середнє відхилення результатів p-value не перевищувало 5%. Пристрій успішно ідентифікував очевидно невипадкові послідовності та підтвердив високу якість випадкових. Надійність результатів підтверджено триразовими проганами з різними послідовностями.

Результати аналізу вказують на високу точність і стабільність пристрою, з ефективною реалізацією логіки обробки даних. Пристрій має низку переваг: автономність, компактність, відсутність залежності від ОС, підтримка збереження результатів, простота використання навіть у неспеціалізованих умовах. У порівнянні з програмними засобами пристрій виграє в мобільності, енергоефективності та зручності використання.

Оцінка якості реалізації довела, що пристрій здатен конкурувати з програмними системами в задачах попереднього, польового або локального контролю генераторів ПВП. У додаткових тестах з обмеженими ресурсами пристрій стабільно працював при зниженій напрузі, високій температурі та

нестабільному живленні, що підтверджує його придатність для вбудованих і критичних середовищ.

Загалом, розроблений пристрій вирішує поставлену задачу – забезпечення автономної, точної та швидкої перевірки псевдовипадкових послідовностей відповідно до міжнародних вимог. Він демонструє практичну придатність до використання в освіті, технічному аудиті, контролі генераторів, тестуванні криптографічних засобів, дослідницьких проєктах, а також у виробничих процесах при сертифікації генераторів ПВП.

Наукова новизна проєкту полягає в поєднанні класичних статистичних алгоритмів із апаратною реалізацією на обмежених ресурсах, що дозволяє досягти високого ступеня мобільності, ефективності та незалежності від зовнішніх програмних середовищ. Розроблений пристрій може слугувати прототипом для майбутніх індустріальних рішень, адаптованих під конкретні прикладні потреби.

У рамках подальших досліджень можливе розширення функціоналу: впровадження нових тестів (Diehard, TestU01), підключення до комп’ютера через USB, додавання екрана з розширеним графічним інтерфейсом, розробка мобільного застосунку для дистанційного контролю, впровадження Wi-Fi/BT модуля для інтеграції з хмарними системами моніторингу.

Підбиваючи підсумок, результати дипломної роботи демонструють успішне вирішення поставленої задачі. Розробка повністю відповідає сучасним вимогам до апаратних засобів тестування ПВП, забезпечує належний рівень точності, автономності, надійності та масштабованості. Її реалізація має практичне значення для галузі інформаційної безпеки та може бути основою для подальших досліджень та комерціалізації.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. NIST Special Publication 800-22. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications / A. Rukhin et al. – Gaithersburg, MD : National Institute of Standards and Technology, 2010. – 131 p.

2. Васильєв, С. А. Криптографія: підручник / С. А. Васильєв. – Харків : ХНУРЕ, 2017. – 328 с.

3. Добровольський, В. І. Інформаційна безпека: основи, методи, технології : навч. посіб. / В. І. Добровольський, І. Б. Романенко. – К. : Кондор, 2021. – 284 с.

4. Cypress Semiconductor. Application Note: PRBS Generators – AN214 [Електронний ресурс]. – Режим доступу: https://www.cypress.com/file/138246/download (дата звернення: 20.04.2025).

5. Програмування мікроконтролерів STM32 на мові С : навч. посіб. / О. В. Гнатюк, І. Ю. Жуков. – Вінниця : ВНТУ, 2020. – 228 с.

6. Гриценко, І. С. Методи оцінювання випадковості у криптографічних застосуваннях / І. С. Гриценко // Збірник наукових праць НТУУ «КПІ». Серія: Інформатика та обчислювальна техніка. – 2022. – № 67. – С. 47–54.

7. STMicroelectronics. STM32F103x8/xB Datasheet – ARM Cortex-M3 32-bit MCU [Електронний ресурс]. – Режим доступу: https://www.st.com/resource/en/datasheet/stm32f103c8.pdf (дата звернення: 22.04.2025).

8. Мельник, Ю. О. Основи проектування цифрових пристроїв : навч. посіб. / Ю. О. Мельник. – Львів : Видавництво Львівської політехніки, 2019. – 312 с.

9. Ахметов, Р. А. Алгоритми і структури даних у мовах С/С++ / Р. А. Ахметов. – Дніпро : Університет «ДНУ», 2021. – 237 с.

10. Чепурний, Д. В. Мікроконтролери AVR: архітектура, програмування, проєктування / Д. В. Чепурний. – Львів : ЛьвДУВС, 2022. – 265 с.

11. Рой, Д. Генерація випадкових чисел і тестування їх якості / Д. Рой, А. Махмуд // Міжнародний науковий журнал «Інформатика та інформаційні технології». – 2021. – № 1(9). – С. 38–44.

12. Python random module documentation [Електронний ресурс] // Python Software Foundation. – Режим доступу: https://docs.python.org/3/library/random.html (дата звернення: 25.04.2025).

13. Сидоренко, М. М. Основи захисту інформації : навч. посіб. / М. М. Сидоренко. – Одеса : ОНУ, 2020. – 198 с.

14. Глушков, В. М. Теорія інформації і кодування / В. М. Глушков. – К. : Наукова думка, 2018. – 364 с.

15. Інститут програмних систем НАН України. Тестування генераторів випадкових чисел: методика та програмні засоби [Електронний ресурс]. – Режим доступу: https://random.ips.org.ua/testsuite (дата звернення: 23.04.2025).

16. Arduino Reference. Pseudo-random Number Generation [Електронний ресурс]. – Режим доступу: https://www.arduino.cc/reference/en/language/functions/random-numbers/random/ (дата звернення: 19.04.2025).

17. Методичні вказівки до виконання дипломної роботи для здобувачів ступеня бакалавра / НТУУ «КПІ». – Київ : ВПІ, 2020. – 24 с.

Додаток А

Зразок реалізації частотного тесту на STM32 (мовою С)

```
#include "math.h"
#include "stdio.h"
#include "string.h"
#include "stdlib.h"

#define BITSTREAM_LENGTH 10000 // Кількість бітів у послідовності

// Припускаємо, що дані зчитані у масив bits[]
uint8_t bits[BITSTREAM_LENGTH];

// Функція частотного тесту
float frequency_monobit_test(uint8_t *bitstream, uint32_t length) {
    int sum = 0;

    for (uint32_t i = 0; i < length; i++) {
        if (bitstream[i] == 1)
            sum += 1;
        else
            sum -= 1;
    }

    float s_obs = fabs(sum) / sqrtf(length);
    float p_value = erfc(s_obs / sqrtf(2));  // Функція додаткової помилки

    return p_value;  // Чим ближче до 1, тим краща випадковість
}

// Виклик функції з демонстрацією результату
void run_frequency_test() {
    // Заповнення test даними (наприклад, випадковими чи з SD-карти)
    for (uint32_t i = 0; i < BITSTREAM_LENGTH; i++) {
        bits[i] = rand() % 2;
    }
```

```
    float result = frequency_monobit_test(bits, BITSTREAM_LENGTH);

    if (result >= 0.01) {
        printf("Тест пройдено. p-значення = %.5f\n", result);
    } else {
        printf("Тест не пройдено. p-значення = %.5f\n", result);
    }
}
```