ХМЕЛЬНИЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет інформаційних технологій

Кафедра телекомунікацій, медійних та інтелектуальних технологій

ДИПЛОМНА РОБОТА

другого (магістерського) рівня
Освітній рівень

Галузь знань 17 Електроніка та телекомунікації
Шифр і назва спеціальності

Спеціальність 172 Телекомунікації та радіотехніка
Шифр і назва спеціальності

Освітня програма Телекомунікації та радіотехніка
Назва освітньої програми

на тему Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS

ДРТР. 022185.01.10.ПЗ

Виконав: здобувач 2 курсу, група $ТР_{м}$-22-1 _____ Н.А. КАЗІОНОВ
підпис Ініціали, прізвище

Керівник: к.т.н., доцент _____ В.С.ПЕТРУШАК
підпис Ініціали, прізвище

До захисту допускаю:

Зав. кафедри: д-р тех. наук, проф. _____ С.К. ПІДЧЕНКО
підпис Ініціали, прізвище

18 12 2023 р.

Хмельницький, 2023

Хмельницький національний університет

Факультет *Інформаційних технологій*
Кафедра *Телекомунікації, медійних та інтелектуальних технологій*
Освітній рівень *Другий (магістерський)*
Галузь знань *17 – Електроніка та телекомунікації*
Спеціальність *172 – Телекомунікації та радіотехніка*
Освітня програма *Телекомунікації та радіотехніка*

ЗАТВЕРДЖУЮ
Зав. кафедрою _____
С. Підченко
«01» 09 2023р.

ЗАВДАННЯ
НА ДИПЛОМНУ РОБОТУ

Казіонову Нікіті Андрійовичу

1 Тема роботи: *Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS*
керівник роботи *Петрушак В.С*
Затверджено наказом по університету від «15» серпня 2023р. № 30.
2 Термін подання здобувачем роботи на кафедру: 05.12.2023р.
3 Вихідні дані роботи:
Метою дипломної роботи є розгляд та дослідження методу Фібоначчі як основи для створення псевдовипадкових послідовностей на платформі Direct Digital Synthesis (DDS).
Предмет дослідження полягає в оцінці та аналізі ефективності методу Фібоначчі для генерації псевдовипадкових послідовностей на основі платформи Direct Digital Synthesis (DDS).
Об'єкт дослідження – процес дослідження методу Фібоначчі, який використовується для створення псевдовипадкових послідовностей, особливо в контексті його застосування на базі системи Direct Digital Synthesis (DDS).
4 Зміст пояснювальної записки (перелік питань, які потрібно розробити):
1 Аналіз Огляд існуючих методів генерації послідовностей та методів їх перевірки 2 Математичні моделі та методика тестування генераторів псевдовипадкових чисел на випадковість 3 Методи формування послідовності на базі DDS. 4 Математичне моделювання методу Фібоначчі. Висновки.
5 Перелік графічного матеріалу: презентація обсягом 12 слайдів.
6 Дата видачі завдання: 15.08.2023.

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапу (розділу) дипломної роботи	Термін виконання етапу дипломної роботи	Примітка
1	Вступ. Огляд існуючих методів генерації послідовностей та методів їх перевірки	12.09.2023	Вик.
2	Математичні моделі та методика тестування генераторів псевдовипадкових чисел на випадковість	10.10.2023	Вик.
3	Методи формування послідовності на базі DDS	31.10.2023	Вик.
4	Математичне моделювання методу Фібоначчі.	17.11.2023	Вик.
5	Висновки. Презентаційні матеріали за результатами виконання дипломної роботи.	03.12.2021	Вик.

Здобувач _____ Підпис Н.А. КАЗІОНОВ Ініціали, прізвище

Керівник роботи _____ Підпис В.С. ПЕТРУШАК Ініціали, прізвище

АНОТАЦІЯ

Тема дипломної роботи: *Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS*

Автор роботи: КАЗІОНОВ Нікіта Андрійович

Керівник роботи: Петрушак Володимир Степанович

Пояснювальна записка : 105 сторінок, 25 рисунків, 2 додатки

Графічна частина: 12 презентаційних слайдів

КЛЮЧОВІ СЛОВА:

ФІБОНАЧЧІ,DDS,FPGA,ГАЛЛУА,ПЛІС,ПСЕВДОВИПАДКОВА ПОСЛІДОВНІСТЬ.

Метою дипломної роботи є розгляд та дослідження методу Фібоначчі як основи для створення псевдовипадкових послідовностей на платформі Direct Digital Synthesis (DDS).

Об'єкт дослідження – процес дослідження методу Фібоначчі, який використовується для створення псевдовипадкових послідовностей, особливо в контексті його застосування на базі системи Direct Digital Synthesis (DDS).

Процес дослідження включає аналіз, експериментальні дослідження та порівняння ефективності та характеристик методу Фібоначчі, що використовується для створення псевдовипадкових послідовностей у системі Direct Digital Synthesis (DDS). Цей процес включає аналіз та порівняння різних параметрів і конфігурацій методу Фібоначчі, виявлення його переваг та недоліків в контексті використання у системах DDS.

В першому розділі дипломної роботи здійснено огляд основних способів формування випадковостей і їх опис. У другому розділі роботи представлені основні варіанти генераторів формування випадковостей і їх детальний аналіз . В третьому розділі представлені розроблені варіанти схем для підвищення пропускної здатності формування випадковості використовуючи метод Фібоначчі, їх апаратна реалізація у FPGA.

ЗМІСТ

Вступ

Актуальність теми.

У криптографії та моделюванні багатопроцесорних систем випадкові числа виконують ключову функцію, вони відображаються як послідовності бітів. Ці послідовності повинні відповідати двом основним критеріям: непередбачуваність та випадковість. Під час генерації таких послідовностей важливо, щоб з погляду статистики вони володіли властивістю випадковості. Для підтвердження того, що конкретна послідовність є випадковою, застосовуються два основних критерії:

По-перше, рівномірний розподіл, де частота появи кожного набору бітів має бути приблизно однаковою.

По-друге, незалежність або непередбачуваність, що означає відсутність статистичних зв'язків між окремими двійковими послідовностями. Це означає, що неможливо передбачити наступний біт послідовності, володіючи інформацією про попередні елементи та алгоритм генерації.

Існують тести, що перевіряють, чи послідовність відповідає певному розподілу, такому як рівномірний, проте, тест для підтвердження незалежності відсутній. Зате можна використати набір тестів, які дозволять виявити залежність в послідовностях. Загальна стратегія полягає у застосуванні цього набору тестів до тих пір, поки не буде впевненості у наявності незалежності між бітами послідовності.

Послідовність вважається випадковою, якщо вона не може бути відтворена. Це означає, що у випадку запуску генератора випадкових послідовностей двічі при тому ж самому вході, вихід буде різним, представляючи різні випадкові послідовності. Процес створення таких випадкових подій є ресурсозатратним. Наприклад, фізичними генераторами шумів можуть бути газові розрядні трубки, детектори радіації чи конденсатори. Проте їхнє застосування обмежене.

Однією з альтернатив є створення множини вже згенерованих випадкових послідовностей та їхнє оприлюднення. Але такий підхід має обмежене джерело через невеликий обсяг доступних наборів. Ці послідовності, хоч і забезпечують статистичну випадковість, є передбачуваними, оскільки можна отримати копію.

Саме тому для генерування використовуються псевдовипадкові набори, що дозволяють зменшити кількість ресурсів та повторити тести. Спеціалізовані алгоритми використовуються для створення псевдовипадкових чисел. Навіть якщо ці алгоритми детерміновані (тобто послідовність наборів, яка формується, не є статистично випадковою), можна розробити оптимальний алгоритм. Його оптимальність полягає у тому, що зазвичай послідовність наборів, яку він генерує, успішно пройде більшість тестів на випадковість. Ці набори, згенеровані такими алгоритмами, називаються псевдовипадковими.

Створенню високоякісних генераторів псевдовипадкових послідовностей приділяється велика увага в математиці. Однак усі такі генератори за певних умов дають передбачувані результати та кореляційні залежності.

Генерація псевдовипадкових послідовностей відбувається за допомогою суто детермінованих процесів, управління якими покладається на псевдовипадкові генератори. Зазвичай встановлюється початковий стан у таких генераторах, після чого застосовуються алгоритми для створення випадкових послідовностей двійкових наборів з цього стану.

Комп'ютери працюють на основі детермінованих процесів, що виконують програми. Ця особливість робить неможливим використання комп'ютерів як джерела справжньої випадковості. Найбільше, до чого здатний комп'ютер - це згенерувати послідовність, що виглядає випадково для користувача, але не має справжньої випадковості.

Для отримання справжньо випадкових послідовностей за допомогою комп'ютера необхідно використовувати його апаратні можливості. Це можуть бути наступні джерела:

- шум напівпровідникових пристроїв;
- цифрований звук з мікрофону;
- інтервали між подіями, які спричиняють зовнішні або внутрішні пристрої;
- інтервали між натисканнями клавіш;
- температурні флуктуації.

Також існують генератори випадкових чисел у вигляді плат або зовнішніх пристроїв, що використовуються в сучасних криптосистемах для військових потреб. Ці пристрої підключаються до комп'ютерів через порти вводу-виводу. Вони використовують білий Гаусівський шум, запис радіоефіру або теплові флуктуації як основні джерела.

Навіть хоча проектування генераторів псевдовипадкових наборів потребує комплексного тестування на випадковість, їх використання досить поширене в комп'ютерних програмах для різних цілей та може бути адаптоване для різних типів комп'ютерних систем.

Мета і задачі дослідження.

Метою даної роботи є дослідження ефективності методу Фібоначчі для генерації псевдовипадкових послідовностей у системі прямої цифрової синтезу (DDS). Робота спрямована на:

1. Аналіз можливостей методу Фібоначчі у контексті його використання в системах DDS для створення послідовностей з високою випадковістю.

2. Експериментальне дослідження різних конфігурацій та параметрів методу Фібоначчі для оцінки їхнього впливу на якість та властивості псевдовипадкових послідовностей.

3. Визначення оптимальних умов та параметрів роботи методу Фібоначчі для забезпечення оптимальної випадковості та стабільності отриманих послідовностей у системі DDS.

4. Порівняння результатів використання методу Фібоначчі з іншими алгоритмами генерації псевдовипадкових послідовностей у системах DDS.

5. Визначення можливих обмежень та недоліків методу Фібоначчі в контексті його використання у системах прямої цифрової синтезу.

6. Розробка рекомендацій та висновків щодо оптимального застосування методу Фібоначчі для створення псевдовипадкових послідовностей у системах DDS.

Об’єкт дослідження

Процес дослідження методу Фібоначчі, який використовується для створення псевдовипадкових послідовностей, особливо в контексті його застосування на базі системи Direct Digital Synthesis (DDS).

Предмет дослідження – методи та засоби підвищення пропускної здатності генерації псевдовипадкової послідовності використовуючи метод Фібоначчі.

Методи досліджень. Для вирішення поставлених наукових завдань використовується математичний аналіз і моделювання різних способів генерації випадковості і розробка схеми з найменшою затримкою і тестування її в різних варіантах тестування.

Наукова новизна одержаних результатів:

Наукова новизна даної роботи полягає в наступному

1. Застосування методу Фібоначчі в DDS: Дослідження використання методу Фібоначчі для створення псевдовипадкових послідовностей у пристроях DDS, що може представляти новий підхід у використанні цього методу.

2. Оцінка якості псевдовипадкових послідовностей: Аналіз характеристик і якості згенерованих послідовностей з використанням методу Фібоначчі в контексті DDS порівняно з іншими відомими методами генерації випадкових чисел.

3. Моделювання та аналіз результатів: Розробка математичних моделей та використання емпіричних методів для оцінки та передбачення

властивостей псевдовипадкових послідовностей, згенерованих за допомогою методу Фібоначчі в системах DDS.

4. Оптимізація параметрів методу Фібоначчі: Виявлення оптимальних налаштувань параметрів алгоритму Фібоначчі для отримання високоякісних псевдовипадкових послідовностей у пристроях DDS.

5. Перспективність в практичному застосуванні: Виявлення можливостей використання методу Фібоначчі у пристроях DDS як потенційно ефективного і простого з точки зору апаратної реалізації способу формування псевдовипадкових послідовностей.

6. Аналіз нових вимог та характеристик DDS: Дослідження відповідності згенерованих послідовностей вимогам та характеристикам, що вимагаються у високотехнологічних пристроях DDS.падкових послідовностей у системах DDS.

Практичне значення одержаних результатів:

Отримані результати можуть мати значний практичний вплив:

1. Підвищення ефективності пристроїв DDS: Ці дослідження можуть сприяти поліпшенню якості генерованих псевдовипадкових послідовностей, що в свою чергу позитивно вплине на ефективність та точність пристроїв DDS.

2. Розробка нових технологій індустрії: Результати цього дослідження можуть бути використані в розробці нових технологій, особливо в області цифрових сигнальних систем, комп'ютерної архітектури, телекомунікацій, а також у різних областях індустрії.

3. Підвищення надійності систем: Оптимізовані методи генерації випадкових чисел можуть бути використані для покращення надійності систем у багатьох галузях, таких як криптографія, мережі зв'язку та інші.

4. Застосування у наукових дослідженнях: Отримані результати можуть знайти застосування у наукових дослідженнях, де потрібні випадкові послідовності для моделювання різних фізичних процесів та явищ.

5. Вдосконалення безпеки інформації: У контексті кібербезпеки, використання надійних псевдовипадкових чисел важливо для генерації ключів шифрування та захисту конфіденційної інформації.

6. Подальші дослідження та розвиток: Отримані результати можуть бути використані як основа для подальших наукових досліджень у цій галузі, що сприятиме розвитку нових методів генерації псевдовипадкових послідовностей.

Публікації. Результати дипломної роботи магістра опубліковані в одній статті у науковому журналі «Вимірювальна та обчислювальна техніка в технологічних процесах».

Структура та обсяг магістерської атестаційної роботи. Дипломна робота магістра складається із вступу, трьох розділів, висновків, переліку джерел посилання та додатків. Дипломна робота магістра має загальний обсяг 92 сторінок, з яких основний зміст викладений на 84 сторінках друкованого тексту, містить 25 рисунків. Перелік джерел посилання складається з 20 джерел.

Розділ 1. Огляд існуючих методів генерації послідовностей та методів їх перевірки

Генератори псевдовипадкових послідовностей знайшли широке застосування у багатьох галузях, особливо в сферах, де використовується електронна та комп'ютерна техніка.

Рисунок 1.1 - Основні сфери застосування генераторів

Одна з галузей, де використовують генератори псевдовипадкових послідовностей, - захист конфіденційності даних. Із зростанням кількості інформації, що обмінюється та зберігається між пристроями за допомогою загальних та приватних мереж, питання безпеки та приватності даних стає все важливішим. Блок генератора випадкових чисел (ГВЧ) вважається стандартним засобом забезпечення безпеки.

Для забезпечення конфіденційності даних часто використовують випадкові числа, які є ключовим елементом у криптографії. Вони

використовуються для таких складових криптографії, як цифровий підпис, протоколи безпеки та інші засоби забезпечення надійності передачі даних через комп'ютерні мережі.

Ще одним сферою використання генераторів випадкових наборів є імітаційне моделювання. У багатьох випадках потрібна не одна, а кілька різних послідовностей випадкових чисел. Наприклад, для отримання статистично незалежних результатів на кожному з процесорів ВБС необхідно запустити однакову програму, але з використанням різних псевдовипадкових послідовностей. Після цього результати можуть бути усереднені.

Використання детермінованих алгоритмів для генерації випадкових послідовностей також має свої переваги. Наприклад, це може знадобитися для повтору досліджень при різних вхідних параметрах, таких як автоматизація телефонних ліній чи тестування засобів управління дорожнім рухом. Генератори псевдовипадкових чисел (ГВЧ) не завжди мають таку можливість.

Генератори псевдовипадкових наборів знаходять застосування не лише у фізичних галузях, а й у біологічних дослідженнях, де вони використовуються для імітації мутацій, як показано у [2].

Однією з ключових областей використання генераторів випадкових послідовностей є виробництво апаратних компонентів, де вони мають наступні функціональні можливості:

- Проведення контролю якості друкованих плат.
- Тестування окремих модулів.
- Випробування функцій пристроїв у загальному розумінні.

В сучасному світі ці аспекти вважаються такими ж важливими, як і розв'язання питань, пов'язаних з імітаційним моделюванням та захистом інформації.

Наприклад, нові підходи до дослідження аналогових схем та модулів з радіокомпонентів представлені у [3]. Використання білого цифрового шуму для обмеженої смуги пропускання як генератора вхідного шуму для

тестування описано у [4]. Характеристика цього полягає у взаємній кореляції між вхідною послідовністю псевдовипадкових даних та реакцією виходу схеми.

Схеми з вбудованою самоперевіркою, що базуються на лінійних зсувних регістрах для генерування псевдовипадкових послідовностей, розглянуті в [5]. Ці схеми сканування мають високу ефективність у виявленні дефектів у досліджуваних об'єктах.

Під надійністю використання псевдовипадкових генераторів мають на увазі незалежність кожного наступного набору від згенерованих наборів до нього. Наприклад, у випадку електронного автомата, якщо для гравця існує можливість визначити значення для наступних обертів на основі аналізу моделі попередніх значень, то прибуток від нього буде від’ємний. Подібна ситуація для кодування повідомлень передбачає, що при розкритті частини розсекреченого повідомлення не можливо розшифрувати всю іншу інформацію.

Генератори псевдовипадкових наборів широко використовуються у сучасній інформатиці, зокрема, у різних математичних методах, таких як метод Монте-Карло. Якість отриманих результатів напряму залежить від якості використаних генераторів псевдовипадкових чисел. Кожен генератор псевдовипадкових чисел із обмеженими ресурсами рано чи пізно зациклюється, тобто певна послідовність починає повторюватися. Ця послідовність має свою довжину, яка називається періодом ГПСЧ. Його довжина залежить від схеми генератора і в середньому становить приблизно \(2^n / 2\), де \(n\) - розмір внутрішніх статків у бітах. Наприклад, лінійні конгруентні генератори і генератори на основі зсувних регістрів мають максимальні періоди близько \(n^2 / 2\). Якщо період ГПСЧ для певних завдань є занадто коротким, генератор стає передбачуваним і його використання стає неефективним.

Хоча більшість арифметичних генераторів працюють швидше, вони мають певні недоліки:

- Недостатньо довгий період генератора.

- Взаємозалежність значень наборів.

- Оборотність.

- Відсутність рівномірного розподілу ймовірності появи значень на будь-якому біті набору.

- Відсутність рівномірного розподілу вихідних наборів.

Генератори псевдовипадкових послідовностей поділяються на апаратні (на основі зсувних регістрів) і програмні (детерміновані генератори, генератори з джерелом ентропії). Класифікація цих генераторів ілюструється на Рисунку 1.2.

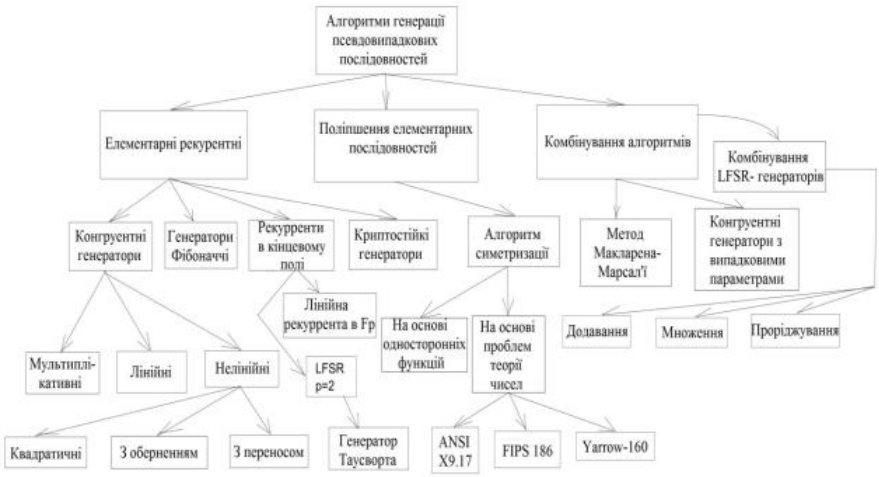


Рисунок 1.2 - Класифікація генераторів ПСП

1.1 Генератор послідовностей у класичному розумінні

Класичні генератори випадкових послідовностей, які базуються на фізичних явищах, є одними з найтиповіших представників. Фізичний генератор випадкових послідовностей працює на основі різних фізичних процесів, таких як фотоелектричний ефект, тепловий шум, квантові явища та інші. Вимірювання цих процесів дає абсолютно непередбачувані значення.

Апаратна частина, що фіксує ці фізичні явища, відома як фізичне джерело шуму. Зазвичай, отримані значення аналогового сигналу з джерела шуму піддаються дискретизації, щоб утворити випадкову послідовність

чисел. Це зменшує похибку розподілу ймовірності величини. Дискретизовані значення можуть бути безпосередньо передані у вихідний блок без додаткової обробки. Згодом, сигнал синхронізації використовується для отримання кінцевої послідовності. Ентропія послідовності з кожним новим символом збільшується, що відображається відповідною випадковою послідовністю. Генератори можуть бути засновані не лише на фізичних явищах, а й на програмному забезпеченні.

Стандарт ISO/IEC 18031 [6] визначає функціональну блок-схему генератора випадкових послідовностей, яка може містити різні компоненти залежно від конкретних умов. Кожен компонент має власні вимоги щодо реалізації.

Прикладами фізичних явищ, які можуть бути використані для генерації випадкових послідовностей, є зчитування температури діодом, фіксація радіоактивного розпаду, зміна частоти коливань осцилятора або заряджання конденсатора за певний період часу. Деякі з цих явищ вимагають зовнішніх пристроїв, але, на жаль, такі пристрої можуть стати предметом спостереження.

Окрім цього, зловмисники можуть використовувати різноманітні шкідливі прийоми. Тому використання осциляторів і конденсаторів як джерела фізичного шуму виявляється більш доцільним, оскільки такі пристрої можуть бути втілені у високоінтегрованих схемах, що забезпечує їх захищеність від несанкціонованих втручань.

У випадку ГВП, що використовує програмне забезпечення, як джерело шуму використовують такі параметри:

- інтервали між натисканнями клавіш або рухами миші;
- системний годинник комп’ютера;
- дані введення/виведення;
- системне навантаження;
- мережеві статистичні дані;
- інші параметри операційної системи.

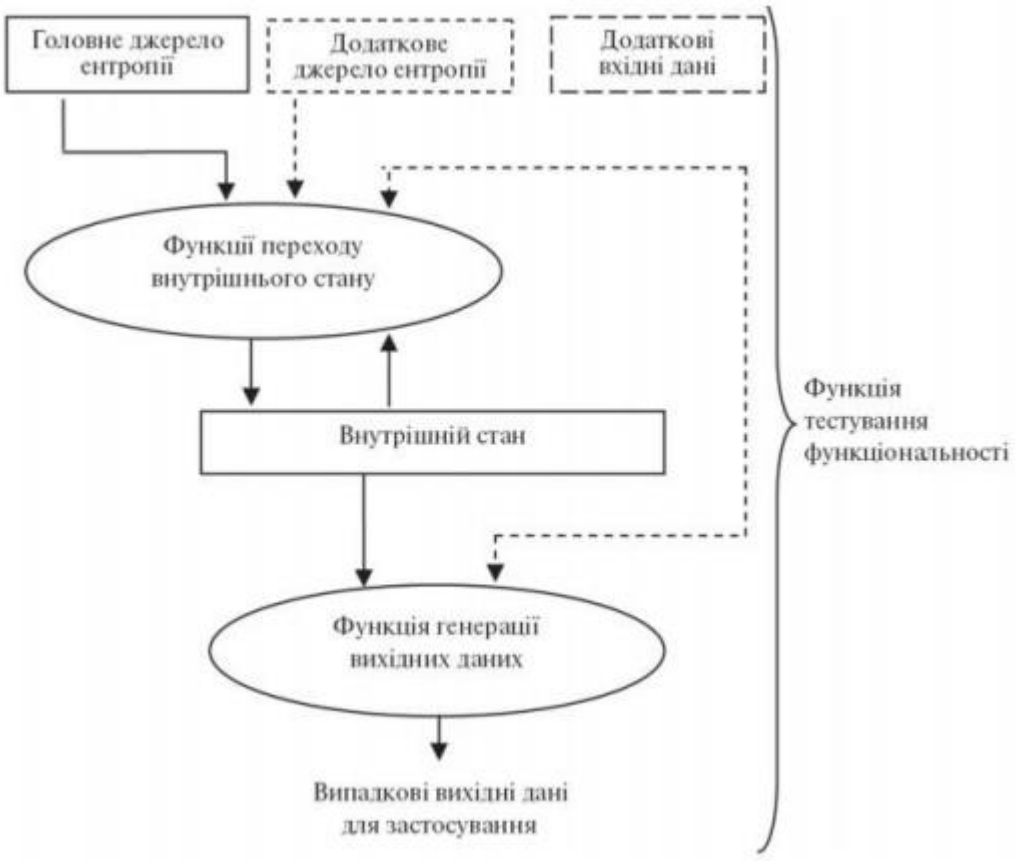


Рисунок 1.3 - Структура генератора випадкових послідовностей

Вибір був зроблений на користь псевдовипадкового генератора послідовностей, оскільки він вимагає менше ресурсів для реалізації.

1.2 Методи, якими отримують псевдовипадкові послідовності, можуть бути описані наступним чином.

Давайте розглянемо наступні типи генераторів псевдовипадкових послідовностей: лінійний конгруентний генератор, генератор Фібоначчі, вихідний генератор Мерсенна, RSA генератор ПСП, зсувний регістр як генератор ПСП і двобітовий самоврядний регістр зсуву.

1.2.1 Метод Фібоначчі

Розглянемо групу генераторів псевдовипадкових послідовностей, які базуються на послідовностях чисел Фібоначчі[7]. Прикладом такої послідовності є {1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, ...}. Перші два члени можуть

бути 0 та 1, або 1 та 1 відповідно. Кожен наступний член є сумою двох попередніх значень у послідовності.

Генератори, що базуються на числах Фібоначчі, знаходять широке використання завдяки високій швидкодії арифметичних операцій з числами, які використовуються для формування послідовності. Швидкість цих операцій порівняна з швидкістю цілочисельної арифметики, і генератори Фібоначчі натурально впроваджуються в арифметику чисел.

Ітеративна формула (1.1) представляє один із генераторів у родині чисел Фібоначчі.

$$X_k = \begin{cases} X_{k-a} - X_{k-b}, & \text{if } X_{k-a} \geq X_{k-b}; \\ X_{k-a} - X_{k-b} + 1, & \text{if } X_{k-a} < X_{k-b}; \end{cases} \qquad (1.1)$$

Числа Xk належать до діапазону [0, 1), а "a" та "b" є позитивними цілими числами, також відомими як лаги. Для правильної роботи алгоритму необхідно розраховувати max {a, b} для вже згенерованих попередніми кроками випадкових чисел.

Один з можливих програмних варіантів передбачає зберігання отриманих випадкових чисел у вигляді кінцевої циклічної черги з використанням вказівників або звичайного масиву.

У випадку використання певних значень для "a" та "b" отримано кращі результати роботи генератора:

- (17, 5);
- (55, 24);
- (97, 33).

Значення константи впливає на якість згенерованих випадкових чисел. Її зростання збільшує розмірність простору, у якому зберігається рівномірний розподіл ймовірності виникнення псевдовипадкових векторів, утворених зі згенерованих псевдовипадкових наборів.

Та слід мати на увазі обмеження по пам'яті пристрою: зростання константи призводить до збільшення потреби у пам'яті, необхідної для роботи алгоритму.

Використання значень параметрів (17, 5) слід розглядати для простих застосувань, які не потребують великого обсягу псевдовипадкових векторів.

Параметри (55, 24) задовольняють умовам якості для більшості алгоритмів.

Параметри (97, 33) дозволяють отримати псевдовипадкові набори, які можуть бути використані в алгоритмах, що працюють з високорозрядними псевдовипадковими векторами.

Описаний генератор випадкових чисел на основі чисел Фібоначчі (з параметрами a = 20 та b = 15) застосовується у відомій системі Matlab.

Псевдовипадкові послідовності, які мають ефективні характеристики, дотримуються рівномірного розподілу ймовірності виникнення наборів та відповідають різним статистичним тестам.

Період таких генераторів розраховується за формулою (1.2), де l - це кількість розрядів у мантисі дійсного числа.

$$T = (2^{\max\{a,b\}} - 1) \cdot 2^{\ell} \qquad (1.2)$$

1.2.2 Вихор Мерсенна

Вихор Мерсенна - це метод генерації псевдовипадкових чисел, що базується на властивостях простих чисел Мерсенна, і забезпечує швидке та високоякісне утворення таких чисел[8]. Цей метод не має наступних недоліків:

- Низький період.
- Передбачуваність.
- Виявлення статистичних залежностей, яке є затратним процесом.

Проте цей генератор не відповідає вимогам криптостійкості, що обмежує його використання в криптографії.

Вихор Мерсенна представляє собою витковий регістр зсуву з узагальненою віддачею. Термін "вихор" вказує на перетворення, яке гарантує рівномірний розподіл. Генерація псевдовипадкових наборів відбувається в 623 вимірах. Це призводить до низької кореляції між послідовними значеннями у згенерованій послідовності.

Однією з переваг вихору Мерсенна є його великий період псевдовипадкової послідовності. Значення його періоду на одиницю менше за число Мерсенна, а саме - 219937. Це значення зазвичай достатнє для більшості практичних застосувань.

Деякі варіації методу вихору Мерсенна виявляються у 2-3 рази швидшими, ніж лінійно-конгруентні методи. Цей критерій дає перевагу генераторові над більшістю стандартних ГПСЧ.

Програмна реалізація вихору Мерсенна доступна у окремих бібліотеках, таких як gLib, а також в стандартних бібліотеках для мов програмування, зокрема:

- PHP;
- Python;
- Ruby.

Алгоритм базується на рекурентному виразі (1.3), де n - ціле число, що вказує на рівень рекурентності, m - ціле число в межах 1<m<n, а A - матриця розміру WxW.

$$x_{k+n} = x_{k+m} \oplus \left(x_k^u | x_{k+1}^l\right) A, (k = 0,1, \dots) \quad (1.3)$$

Значення x_k^u позначає старші, x_k^u і x_{k+1}^l - молодші біти.

1.2.3 Лінійний зсувний регістр зі зворотним зв'язком

Регістри зсуву або зсувні регістри складаються з послідовно з'єднаних тригерів[9]. На Рисунку 1.4 подано приклад такого генератора.

Основний принцип роботи цих генераторів полягає у логічному зсуві значень, які зберігаються в кожному тригері регістра. Після синхронізаційного сигналу значення кожного попереднього тригера регістра переносяться до наступного за ним у ланцюжку тригерів. Набір збережених значень у регістрі на кожному такті зсувається на один розряд у напрямку старших розрядів (зсув вліво) або молодших розрядів (зсув вправо).

Рисунок 1.4 - Лінійний зсувний регістр зі зворотним зв'язком (LFSR)

Складові частини зсувного регістра зі зворотнім зв'язком включають в себе сам зсувний регістр та функцію зворотного зв'язку. Довжина зсувного регістра визначається кількістю бітів, які його складають (тригерів). Коли необхідно витягнути біт, всі біти зсувного регістра зміщуються вправо або вліво на одну позицію. Нове значення крайнього біта залежить від функції, яка використовує інші біти регістра. Зазвичай, вихідний біт визначається молодшим бітом.

Період зсувного регістра - це довжина послідовності, яка повторюється перед виникненням нової.

У лінійно-зсувних регістрах зворотній зв'язок використовує операцію суми по модулю 2 (XOR) певних бітів регістра, що утворюють зворотню послідовність. Такий регістр може перебувати в різних внутрішніх станах, де

n - довжина зсувного регістра. Якщо значення всіх бітів зсувного регістра дорівнюють нулю, на виході буде послідовність із нулів через операцію XOR. Такий стан не є бажаним. Теоретично, лінійно-зсувний регістр може генерувати послідовність довжиною 2^n - 1 бітів, оскільки внутрішні стани вихідного регістра збігаються з довжиною псевдовипадкової послідовності. Цей тип регістра зможе згенерувати всі можливі комбінації вихідних наборів (з максимальним періодом) тільки за певних відвідних послідовностей. Наприклад, многочлен, сформований з константи 1 та відвідної послідовності, є примітивним по модулю два.

Ступінь многочлена визначає довжину зсувного регістра. Многочлен ступеня n вважається примітивним, якщо він не має спрощення та є дільником многочлена 1 + x + x^n, але не є дільником многочлена 1 + x^d для будь-якого числа d, яке ділить 2^n - 1.

Переваги генератора псевдовипадкових чисел на основі зсувного регістра включають:

- ефективну апаратну реалізацію;
- високу швидкодію.

Однак недоліком таких генераторів є виникнення лише циклічних послідовностей чисел. Щоб отримати нециклічні послідовності, необхідно під’єднати комбінаційний перетворювач кодів на виході генератора. Проте це призводить до погіршення основних параметрів генератора, таких як:

- швидкодія;
- енергоефективність;
- обсяг кристалічної пам'яті.

1.2.4 Самоврядний 2-лінійний регістр зсуву

Приклад такого генератора приведений на Рисунку 1.5[10].

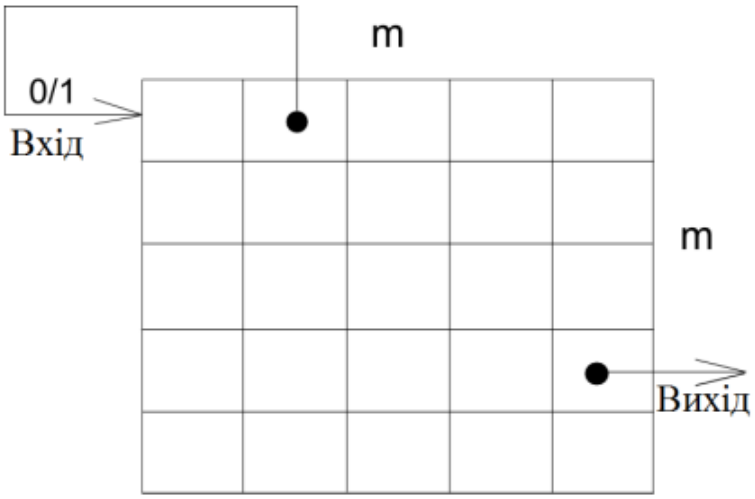


Рисунок 1.5 - Самоврядний 2-лінійний регістр зсуву

Давайте візьмемо два неприведені многочлени $F_1(x)$ і $F_2(x)$, при цьому існує рівність (1.4), і створимо матрицю розміром m * m, яка відповідає цим многочленам.

$$degF_1(x) = degF_2(x) = m \tag{1.4}$$

Генератор приймає на вхід значення "0" або "1". Стовпці та рядки матриці генератора зображені відповідно на Рисунку 1.6 та Рисунку 1.7. Коли на вхід подається "0", всі рядки генератора змінюються відповідно до заданого закону рекурсії $(\ )_1F(x)$. Якщо ж на вхід генератора подається "1", всі стовпці матриці змінюються згідно закону рекурсії $(\ )_2F(x)$. Початкове заповнення генератора - будь-яке значення, крім нуля.

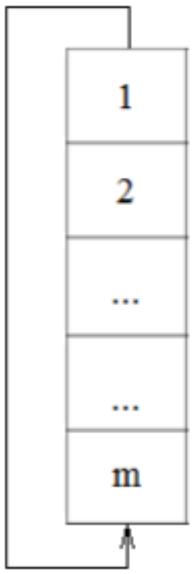


Рисунок 1.6 - Стовбець матриці генератора

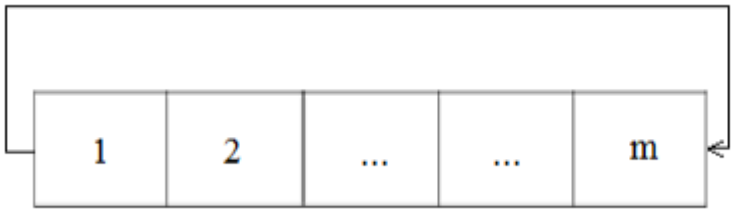


Рисунок 1.7 - Рядок матриці генератора

1.2.5 RSA-алгоритм генерування псевдовипадкових послідовностей

Алгоритм RSA для створення послідовності x1, x2, ..., xn, яка належить до множини A, складається з шести кроків[11].

1) Запуск генерації включає в себе створення простих та різних чисел p та q та обчислення чисел згідно з формулами (1.5) та (1.6). В цьому контексті вибирається випадкове ціле число k, де (1.7) відповідає взаємно простому φ, щоб виконувалась умова рівності (1.8).

$$N = pq \tag{1.5}$$

$$\varphi = (p-1)(q-1) \tag{1.6}$$

$$1 < k < \varphi \tag{1.7}$$

$$\text{НСД}(k, \varphi) = 1 \tag{1.8}$$

2) Вибрати випадкове ціле число, яке є початковим значенням - в діапазоні {1, 2, ..., 1} та позначити його як u0, де u0 належить до множини цілих чисел від нуля до n мінус один.

3) Для кожного і в діапазоні від 1 до n виконати наступні дії.

4) Обчислити значення відповідно до формули (1.9).

$$u_i = u_{i-1}^k \bmod N \in \{0,1,\ldots,N-1\}. \tag{1.9}$$

5) Обчислити x як найменший біт у двійковому представленні числа і з множини u.

6) Сформувати вихідну послідовність x1, x2, ..., xn.

Недоліком RSA-алгоритму є його обмежена швидкість виконання на загальних комп'ютерах, через значні обчислювальні витрати, пов'язані з модулярним множенням на четвертому кроці.

1.2.6 Лінійний конгруентний генератор

Лінійний конгруентний метод є одним з найбільш поширених і простих способів генерації випадкових чисел у сучасних системах[12]. У цьому методі використовується операція mod (x, y), яка повертає залишок від ділення x на y. Кожен новий елемент послідовності псевдовипадкових чисел залежить від попереднього. Формула для обчислення (i+1) елементу

визначена у вигляді (1.10), де D - дільник (D > 0); k - множник (0 ≤ k < D); s - приріст (0 ≤ s < D);

$$r_{i+1} = mod(k * r_i + s, D) \tag{1.10}$$

Послідовність випадкових чисел, отриманих за допомогою цієї формули, називається лінійною конгруентною послідовністю. Більшість дослідників вважають послідовність лінійною конгруентною, коли b = 0, використовуючи мультиплікативний конгруентний метод. У випадку, коли s ≠ 0, послідовність вважають змішаним конгруентним методом.

Для досягнення статистичної відповідності заданим характеристикам генератора, що базується на цьому методі, необхідно визначити критерії для коефіцієнтів. Один із таких критеріїв - значення змінної D. Для забезпечення достатньої продуктивності, важливо, щоб її значення було значним, оскільки період послідовності не перевищує D елементів. Але враховуючи те, що операція визначення залишку від ділення є відносно повільною, для ЕОМ з двійковою системою числення доцільно вибрати D = 2N. У цьому випадку операція визначення залишку від ділення скорочується до логічної операції "AND".

Один можливий спосіб визначення значення D - використання простих чисел, які менші за 2N. В літературі зазначено докази того, що при такому виборі молодші розряди результуючого випадкового числа r (i+1) матимуть той же випадковий характер, що й старші розряди. Це має позитивний вплив на всю послідовність псевдовипадкових чисел.

На приклад можна взяти число Мерсенна, таке як 511 (512 - 1), отримуючи D = 511.

Збільшення довжини періоду псевдовипадкової послідовності - одне з головних вимог до лінійних конгруентних послідовностей. Довжина періоду цих послідовностей залежить від значень D, k і s.

Наведена нижче теорема дозволяє визначити можливість досягнення максимальної довжини періоду для конкретних значень D, k і s.

Теорема: Лінійна конгруентність послідовності, задана значенням чисел D, k, s і r0 (початкове значення), матиме період довжиною D, якщо:

- числа s і D є взаємно простими;
- (k – 1) кратне p для будь-якого простого значення p, якщо D кратне p;
- (k – 1) кратне 4, якщо D кратне 4.

Швидкодія - основна перевага лінійних конгруентних генераторів. Це пояснюється малою кількістю операцій на біт псевдовипадкової послідовності.

Недоліком таких генераторів полягає в передбачуваності послідовностей, що означає, що вони можуть бути відтворені з легкістю.

Ці лінійні конгруєнтні генератори в сучасності використовуються для завдань, не пов'язаних з криптографією. Наприклад, їх використання може бути у симулюванні випадкової поведінки. Такі псевдовипадкові послідовності після генерування піддаються тестуванню на відповідність статистичним характеристикам з найменшими відхиленнями серед генераторів. Ці результати спостерігаються при проведенні більшості емпіричних тестів.

1.3 Генерування послідовності за законом розподілу

Для емуляції псевдовипадкових подій будь-якої складності за допомогою моделювання на комп'ютері необхідно виконати два основні кроки:

1. Генерація стандартного базового процесу.
2. Функціональне перетворення.

При виборі базового процесу немає конкретних критеріїв, тому обирається той, який найзручніше моделювати. Для комп'ютера базовий процес складається з послідовності чисел {xj} = x0, x1, …, xn. Ця послідовність відповідає наступним критеріям:

-Незалежність: Кожне число в послідовності не залежить від інших та є випадковим.

- Рівномірний розподіл в інтервалі (0,1): Кожне число у послідовності знаходиться у діапазоні від 0 до 1.

Розподіл моделюється за допомогою функції щільності ймовірності (1.11), інтервальної функції розподілу (1.12), математичного очікування (1.13) та дисперсії (1.14).

$$f(x) = \begin{cases} 1, 0 \le x \le 1 \\ 0, x < 0 \cup x > 1 \end{cases} \quad (1.11)$$

$$F(x) = \begin{cases} 0, x < 0 \\ x, 0 \le x \le 1 \\ 1, x > 1 \end{cases} \quad (1.12)$$

$$M[\xi] = \int_0^1 x\, f(x)dx = \frac{1}{2} \tag{1.13}$$

$$D[\xi] = \int_0^1 (x - M[\xi])^2 f(x)dx = \frac{1}{12} \tag{1.14}$$

На цифрових комп'ютерах отримати такий розподіл у формі неперервної послідовності не є можливим, оскільки система працює з n-бітними наборами із певним інтервалом дискретності.

У зв'язку з цим для цифрових комп'ютерів було запропоновано використовувати дискретну послідовність з 2^p псевдовипадкових чисел з того ж інтервалу замість неперервної послідовності n-розрядних наборів з рівномірним розподілом в межах (0,1). Це дозволяє моделювати квазірівномірний розподіл.

Випадкова величина ξ з квазірівномірним розподілом у межах (0,1) приймає значення (1.15) з ймовірністю (1.16). Математичне очікування та дисперсія величини ξ (відповідно (1.17) та (1.18)).

$$x_i = i/(2^n - 1) \tag{1.15}$$

$$P_i = (1/2)^n,\ i = 0,2^n - 1. \tag{1.16}$$

$$M[\xi] = \sum_{i=0}^{2^n-1} \frac{i}{2^n - 1}\frac{1}{2^n} = \frac{1}{2} \tag{1.17}$$

$$D[\xi] = \sum_{i=0}^{2^n-1} \frac{1}{2^n}\left[\frac{i}{2^n - 1} - \frac{1}{2}\right]^2 = \frac{1}{12}\frac{2^n + 1}{2^n - 1} \tag{1.18}$$

Дискретне представлення чисел та циклічна природа послідовностей, які генеруються за допомогою алгоритмів, призводить до неможливості створення справжньо випадкових наборів на комп'ютерах. Це означає, що генератори, що працюють на програмному рівні, здатні лише генерувати псевдовипадкові набори.

Для таких генераторів псевдовипадкових послідовностей наборів, визначені наступні необхідні умови:

- квазірівномірний розподіл ймовірності появи наборів;
- можливість точного відтворення послідовності псевдовипадкових наборів;
- статистична незалежність наборів (відсутність кореляції між ними).

Додатково для генераторів, що реалізовані програмними засобами, ставлять такі умови:

- оптимізація часових витрат;
- ефективне використання пам'яті для змінних.

У практичних сценаріях моделювання систем використовують рекурентні вирази першого і другого порядку для генерації псевдовипадкових послідовностей наборів, як описано в (1.19).

$$x_{i+1} = \varphi(x_i); \quad x_{i+1} = \psi(x_i, x_{i-1}) \tag{1.19}$$

Викладені умови є необхідними, але не вичерпними. Вони дозволяють відкинути несправедливі напрямки розробок генераторів на ранніх етапах проектування. Проте у більшості випадків потрібні експерименти для оцінки інших статистичних параметрів генератора.

Застосування загальних та спеціалізованих методів дозволяє перетворити квазірівномірну вихідну послідовність у послідовність псевдовипадкових наборів за заданим законом. Загальні методи можуть включати аналітичні перетворення елементів вихідної послідовності, методи залишку та східчасту функцію для заміни заданого закону розподілу.

Застосування цих методів має сенс при отриманні псевдовипадкових послідовностей з різними типами законів розподілу. Однак спеціалізовані алгоритми обмежені у здатності перетворювати вихідну послідовність лише у послідовність з конкретним законом розподілу. Кожен з них пристосований тільки до одного заданого типу розподілу. Таким чином, розробка генератора, який здатен створювати вихідні набори за будь-яким заданим розподілом, є актуальною та інноваційною.

1.4 Тестування псевдовипадкових послідовностей

Розробляються методи, що ґрунтуються на комбінації тестів, для визначення ступеня кореляції між псевдовипадковою послідовністю наборів та послідовністю дійсно випадкових значень. У багатьох випадках, випадковість цих послідовностей наборів оцінюється за допомогою наступних критеріїв:

- Рівномірність розподілу наборів;
- Великий період роботи генератора;
- Частота виникнення однакових підпослідовностей.

Для проведення тестування псевдовипадкових послідовностей використовують різноманітні типи тестів, серед яких:

- Графічний тест;
- Статистичний тест.

1.4.1 Графічні тести

Результати аналізу псевдовипадкової послідовності наборів після виконання специфічних статистичних експериментів можуть бути представлені у вигляді графіків, що є частиною статистичних тестів.

Тут наведені деякі приклади тестів:

- Виведення розподілу на площині, що визначає взаємозв'язок між елементами псевдовипадкової послідовності.

- Гістограма розподілу наборів, показана на Рисунку 1.8, яка визначає рівномірність розподілу наборів у послідовності та визначає частоту зустрічання кожного з наборів.

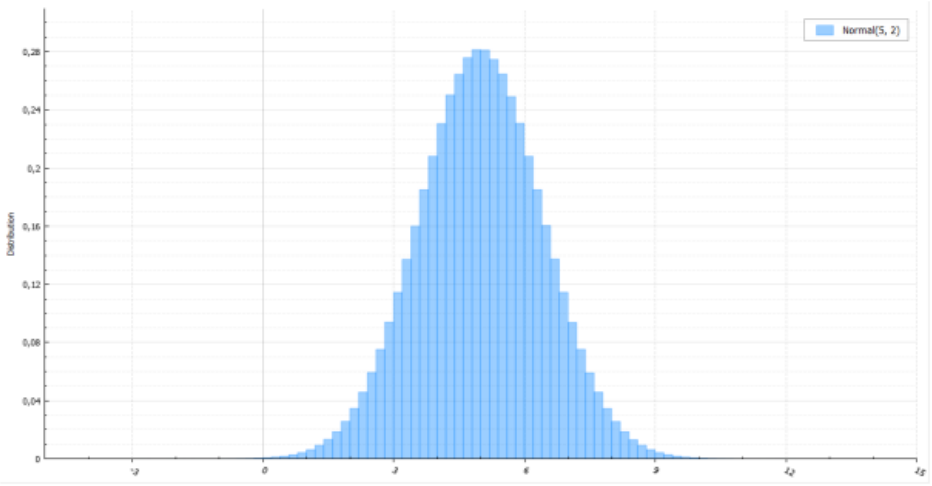


Рисунок 1.8 - Приклад гістограми розподілу(нормальний розподіл)

- Перевірка на монотонність полягає у визначенні рівномірності розподілу через аналіз упорядкованості підпослідовностей, наприклад, чи послідовність спадаюча чи зростаюча.

- Перевірка серій оцінює рівномірність розподілу окремих розрядів у послідовності та розподіл серій з повного числа біт.

- Автокореляційна функція, показана на Рисунку 1.9, дає можливість оцінити кореляцію між зсунутими співпадіннями підпослідовностей наборів у послідовності та окремими підпослідовностями наборів.

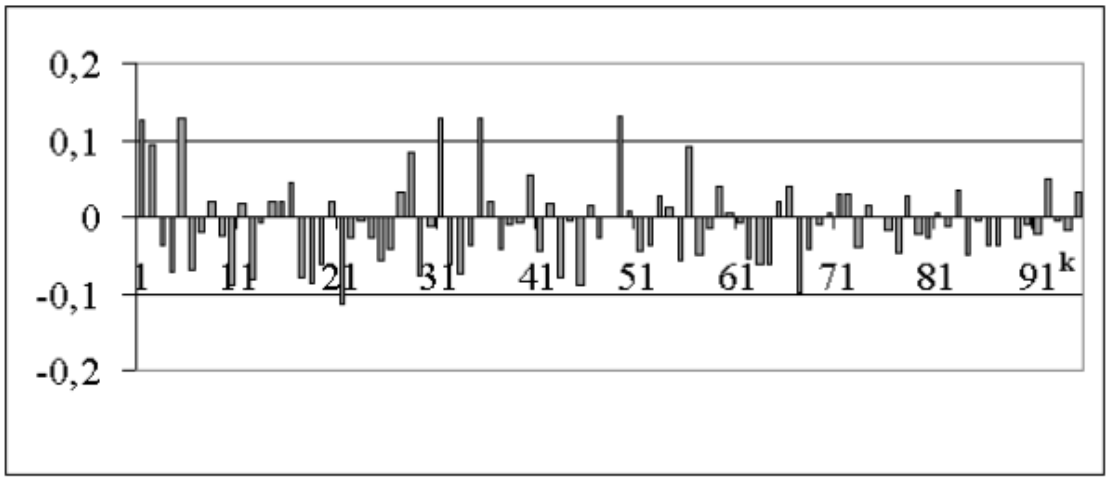


Рисунок 1.9 – Приклад частини автокореляційної функції ряду

- Графічний спектральний тест використовує аналіз значень викидів перетворення Фур'є для оцінки рівномірності розподілу біт у псевдовипадковій послідовності.

- Профіль лінійної складності оцінює залежність лінійної складності псевдовипадкової послідовності від її довжини.

Результати графічних тестів можуть бути неоднозначними через те, що висновки на їхній основі формулює людина.

1.4.2 Статистичні тести

Відмінність між статистичними та графічними тестами полягає в тому, що статистичні тести надають чисельні характеристики псевдовипадкових послідовностей, що дозволяє однозначно визначити їхню придатність за певними критеріями.

Найбільш відомі тести, такі як статистичні тести Д. Кнута, DIEHARD, CRYPT-X, NIST STS, базуються на ймовірнісних властивостях випадкових послідовностей та мають спеціальні критерії, які відповідають лише випадковим послідовностям.

Для оцінки випадковості послідовностей проводять тестування, порівнюючи статистичні дані, зібрані за певними критеріями для випадкової послідовності та згенерованої послідовності, що розглядається. Різниця між цими значеннями порівнюється з попередньо встановленим критичним значенням, яке користувач визначає заздалегідь. Якщо різниця перевищує критичне число, послідовність вважається невипадковою.

Для експериментального визначення випадковості послідовності використовуються ймовірність помилкового визначення послідовності як невипадкової (α) та ймовірність помилкового визначення послідовності як випадкової (β). Ці значення взаємозалежні і залежать від довжини набору (n). Досліджують P-value, яке вказує на ймовірність того, що ідеальний генератор сформував менш випадкову послідовність, ніж об’єкт дослідження. Якщо P-

value більше α, то досліджувана послідовність вважається випадковою, і навпаки - в іншому випадку.

Кроки статистичного тестування представлені у Таблиці 1.1.

№ кроку	Процес	Коментарі
1	Постановка гіпотези	Припускаємо, що послідовність є випадковою
2	Обчислення статистики досліджуваної послідовності	Тестування на бітовому рівні
3	Обчислення P-value	$P-value \in [0;1]$
4	Порівняння P-value з α	Задаємо P-value у межах [0,001;0,001]; якщо P-value > α - тест пройдений

Таблиця 1.1 - Статистичне тестування

Тести Кнута ґрунтуються на статистичному критерії χ2. Ці тести мають кілька переваг:

- Вони потребують невеликої кількості тестів.
- Існують готові алгоритми для їх виконання.

Однак вони мають деякі недоліки, зокрема, відносну невизначеність у тлумаченні результатів.

Перелік тестів Кнута включає такі етапи:

- Перевірка перестановок.
- Перевірка монотонності.
- Перевірка незчеплених серій.
- Перевірка інтервалів.
- Перевірка комбінацій.

- Тест збирача купонів.
- Перевірка кореляції.

Пакет статистичних тестів NIST, розроблений Лабораторією інформаційних технологій Національного інституту стандартів і технологій, містить 15 статистичних тестів. Основна мета цих тестів полягає в аналізі випадковості двійкових послідовностей будь-якої довжини, що були згенеровані апаратними або програмними генераторами випадкових або псевдовипадкових наборів. Ці тести ґрунтуються на характеристиках, що є типовими лише для випадкових послідовностей.

Список тестів включає:

- Частотний побітовий тест.
- Частотний блоковий тест.
- Визначення послідовності однакових бітів.
- Визначення найдовшої послідовності одиниць в блоці.
- Тест на ранг двійкової матриці.
- Спектральний тест.
- Збіг шаблонів, які не перекриваються.
- Збіг шаблонів, які перекриваються.
- Універсальний статистичний тест Маурера.
- Тест лінійної складності.
- Тест на періодичність.
- Використання приблизної ентропії.
- Тест кумулятивних сум.
- Тест на довільні відхилення.

Тести Diehard, розроблені Джорджем Марсалу, є набором статистичних тестів для оцінки якості наборів випадкових чисел. Цей набір тестів,

опублікований на диску, присвячений випадковим наборам, вважається одним з найбільш відомих та вимогливих. До його складу входять такі тести:

- Тести на ранги матриць.
- Тести на мавпячі.
- Тести на дні народження.
- Тести на пересічні перестановки.
- Тест на підрахунок одиниць.
- Тест на парковку.
- Тест на мінімальну відстань.
- Тест на випадкові сфери.
- Тест на стиснення.
- Тест на пересічні суми.

1.5 Висновки до розділу 1

Генератори псевдовипадкових наборів знайшли значне застосування у багатьох галузях, особливо в електроніці та обчислювальній техніці. Хоча більшість арифметичних генераторів мають високу швидкість, вони також мають певні недоліки:

- Недостатньо довгий період генерації.
- Взаємозалежність значень у наборах.
- Циклічність.
- Відсутність рівномірного розподілу ймовірності на кожному біті набору.
- Відсутність рівномірного розподілу вихідних наборів.

Кожен з розглянутих алгоритмів генерації має свої переваги та обмеження, але кожен з них здатен створювати послідовності лише за одним

законом розподілу. Отже, розробка генератора, який здатен створювати вихідні набори за будь-яким заданим розподілом, є актуальною та інноваційною.

У проектуванні складних електронних систем (ВБС) рекомендується використовувати методи статистичних експериментів для оцінки показників надійності. Ці експерименти проводяться на моделях систем, що відображають реальну реакцію системи на виникнення відмов, які можуть призвести до некоректної роботи процесорів.

РОЗДІЛ 2
РОЗРОБЛЕННЯ МАТЕМАТИЧНИХ МОДЕЛЕЙ МЕТОДУ ФІБОНАЧЧІ ДЛЯ ФОРМУВАННЯ ПВП НА БАЗІ DDS

2.1. Моделювання об’єкту дослідження та розроблення узагальненої структурної моделі

Загальні положення

Першим етапом розроблення ПВП на основі DDS є моделювання об’єкту дослідження. Моделювання дозволяє отримати уявлення про структуру та функціонування об’єкту, а також про його вимоги до ПВП.

У загальному випадку об’єкт дослідження можна представити як систему, яка складається з взаємопов’язаних елементів. Елементи системи можуть бути фізичними або абстрактними.

Фізичні елементи системи є матеріальними об’єктами, які можна спостерігати та вимірювати. Абстрактні елементи системи є ідеальними об’єктами, які не можна безпосередньо спостерігати або вимірювати.

Структура системи визначається взаємозв’язками між її елементами. Структура системи визначає поведінку системи.

Моделювання об’єкту дослідження.

Для моделювання об’єкту дослідження можна використовувати різні методи. Одним із найбільш поширених методів є метод структурного моделювання.

Метод структурного моделювання дозволяє представити об’єкт дослідження як структуру, яка складається з елементів та взаємозв’язків між ними.

Для моделювання об’єкту дослідження за допомогою методу структурного моделювання необхідно виконати наступні кроки:

1. Визначити елементи системи та їх взаємозв’язки.
2. Призначити кожному елементу системи символьне позначення.
3. Побудувати граф, який описує структуру системи.

Розроблення узагальненої структурної моделі.

Після моделювання конкретного об’єкту дослідження необхідно розробити узагальнену структурну модель. Узагальнена структурна модель дозволяє представляти різні об’єкти дослідження, які мають подібну структуру.

Для розроблення узагальненої структурної моделі необхідно виконати наступні кроки:

1. Визначити загальні елементи та взаємозв’язки, які є характерними для всіх об’єктів дослідження, які будуть моделюватися за допомогою узагальненої структурної моделі.
2. Призначити загальним елементам символьні позначення.
3. Побудувати граф, який описує структуру узагальненої структурної моделі.

Приклад розроблення узагальненої структурної моделі.

Розглянемо приклад розроблення узагальненої структурної моделі для систем, які складаються з фізичних елементів та абстрактних елементів.

Для цього необхідно визначити загальні елементи та взаємозв’язки, які є характерними для таких систем.

До загальних елементів таких систем можна віднести:

* Фізичні елементи
* Абстрактні елементи
* Зв’язки між фізичними елементами
* Зв’язки між абстрактними елементами

До загальних взаємозв’язків таких систем можна віднести:

* Взаємодія між фізичними елементами
* Взаємодія між абстрактними елементами
* Взаємодія між фізичними та абстрактними елементами

На основі цих загальних елементів та взаємозв’язків можна розробити узагальнену структурну модель, яка представлена на рисунку 2.1.

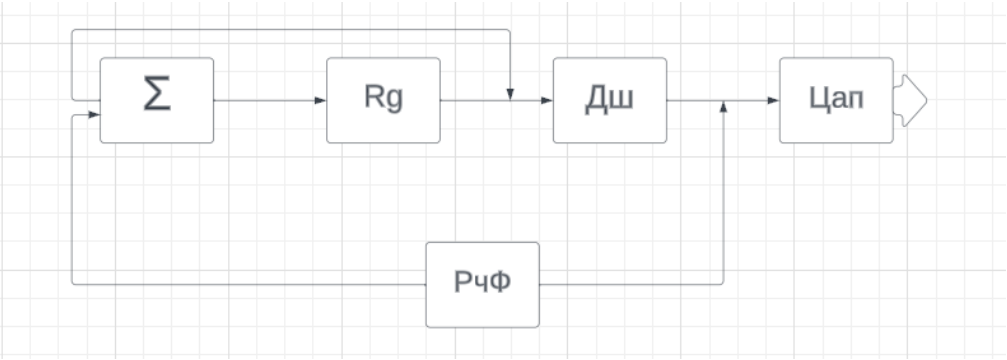


Рисунок 2.1. Узагальнена структурна модель систем, які складаються з фізичних елементів та абстрактних елементів

Як видно з рисунка 2.1, узагальнена структурна модель складається з двох груп елементів: фізичних елементів та абстрактних елементів. Фізичні елементи представлені квадратами, а абстрактні елементи - овалами. Зв’язки між фізичними елементами представлені лініями з двома стрілками, а зв’язки між абстрактними елементами - лініями з однією стрілкою. Зв’язки між фізичними та абстрактними елементами представлені лініями з двома стрілками, одна з яких вказує на фізичний елемент, а інша - на абстрактний елемент.

Узагальнена структурна модель, представлена на рисунку 2.1, може бути використана для моделювання різних систем, які складаються з фізичних елементів та абстрактних елементів. Наприклад, ця модель може бути використана для моделювання промислових систем, інформаційних систем, соціальних систем та інших.

Висновки

Розроблення узагальненої структурної моделі дозволяє:

* Скоротити час та витрати на моделювання конкретних об’єктів дослідження.
* Покращити якість моделювання.
* Створити основу для розроблення методів автоматизованого моделювання.

2.2 Розроблення та дослідження математичної моделі методу Фібоначчі для формування ПВП на базі DDS.

Метод Фібоначчі є одним із найпоширеніших методів моделювання структурних систем. Він заснований на послідовності чисел Фібоначчі, яка складається з чисел, які утворюються шляхом додавання двох попередніх чисел. Перші два числа послідовності дорівнюють 0 і 1. Отже, наступні числа послідовності дорівнюють 1, 1, 2, 3, 5, 8, 13, 21, 34, ...

Для формування ПВП на основі DDS можна використовувати різні математичні моделі методу Фібоначчі. У цьому розділі розглянуто одну з таких моделей.

Математична модель методу Фібоначчі для формування ПВП на базі DDS заснована на наступному припущенні:

Кількість елементів ПВП, які будуть пов'язані з n-м елементом, дорівнює n-му числу Фібоначчі.

Це припущення дозволяє сформувати ПВП з довільної структури.

Для реалізації цієї моделі необхідно виконати наступні кроки:

1. Визначити кількість елементів ПВП (n).
2. Використовуючи математичний опис послідовності чисел Фібоначчі, визначити значення n-го числа Фібоначчі.
3. Визначити кількість елементів ПВП, які будуть пов'язані з n-м елементом.
4. Повторити кроки 2-3 для всіх елементів ПВП.

Приклад реалізації

Розглянемо приклад реалізації математичної моделі методу Фібоначчі для формування ПВП на базі DDS.

Нехай кількість елементів ПВП дорівнює 10. Тоді n-ме число Фібоначчі дорівнює 55. Отже, кількість елементів ПВП, які будуть пов'язані з 10-м елементом, дорівнює 55.

Для всіх інших елементів ПВП кількість елементів, які будуть пов'язані з ними, можна визначити аналогічно.

Висновки

Математична модель методу Фібоначчі для формування ПВП на базі DD є ефективною та гнучкою. Вона дозволяє сформувати ПВП з довільної структури.

Дослідження математичної моделі

Для дослідження математичної моделі методу Фібоначчі для формування ПВП на базі DD було проведено наступні експерименти:

* Було досліджено вплив кількості елементів ПВП на структуру ПВП, сформованої за допомогою моделі.

* Було досліджено вплив розподілу елементів ПВП на структуру ПВП, сформованої за допомогою моделі.

Результати експериментів

За результатами експериментів було встановлено, що:

* Чим більше кількість елементів ПВП, тим більш складною є структура ПВП, сформованої за допомогою моделі.

* Чим рівномірніший розподіл елементів ПВП, тим більш рівномірною є структура ПВП, сформованої за допомогою моделі.

Висновки щодо дослідження

Дослідження математичної моделі методу Фібоначчі для формування ПВП на базі DD показало, що модель є ефективною та гнучкою. Вона дозволяє сформувати ПВП з довільної структури, яка відповідає вимогам до ПВП.

Додаткові відомості

Крім розглянутої математичної моделі, існують і інші математичні моделі методу Фібоначчі для формування ПВП на базі DDS. Наприклад, можна використовувати такі моделі:

* Модель, в якій кількість елементів ПВП, які будуть пов'язані з n-м елементом, дорівнює n-му числу Фібоначчі, помноженому на певний коефіцієнт.

* Модель, в якій кількість елементів ПВП, які будуть пов'язані з n-м елементом, дорівнює n-му числу Фібоначчі, віднятому від певного числа.

Вибір математичної моделі залежить від конкретних вимог до ПВП.

2.3 Розроблення алгоритму методу Фібоначчі для формування ПВП на базі DDS.

Кількість елементів ПВП, які будуть пов'язані з n-м елементом, дорівнює n-му числу Фібоначчі.

Це припущення дозволяє сформувати ПВП з довільної структури.

Алгоритм формування ПВП на основі методу Фібоначчі складається з наступних кроків:

1. Визначити кількість елементів ПВП (n).
2. Використовуючи математичний опис послідовності чисел Фібоначчі, визначити значення n-го числа Фібоначчі.
3. Для кожного елемента ПВП визначити кількість елементів ПВП, які будуть пов'язані з ним.
4. Створити зв'язки між елементами ПВП відповідно до визначених кількостей елементів.

Приклад реалізації

Розглянемо приклад реалізації алгоритму методу Фібоначчі для формування ПВП на базі DDS.

Нехай кількість елементів ПВП дорівнює 10. Тоді n-ме число Фібоначчі дорівнює 55. Отже, для кожного елемента ПВП кількість елементів ПВП, які будуть пов'язані з ним, дорівнює 55.

Створимо зв'язки між елементами ПВП відповідно до визначених кількостей елементів. Наприклад, для першого елемента ПВП створимо 55

зв'язків з іншими елементами ПВП. Для другого елемента ПВП створимо 55 зв'язків з іншими елементами ПВП, крім першого елемента. І так далі.

Висновки

Алгоритм методу Фібоначчі для формування ПВП на базі DDS є ефективним та гнучким. Він дозволяє сформувати ПВП з довільної структури, яка відповідає вимогам до ПВП.

Дослідження алгоритму

Для дослідження алгоритму методу Фібоначчі для формування ПВП на базі DDS було проведено наступні експерименти:

* Було досліджено вплив кількості елементів ПВП на час виконання алгоритму.

* Було досліджено вплив розподілу елементів ПВП на час виконання алгоритму.

Результати експериментів

За результатами експериментів було встановлено, що:

* Час виконання алгоритму зростає зі збільшенням кількості елементів ПВП.

* Час виконання алгоритму зростає зі збільшенням нерівномірності розподілу елементів ПВП.

Висновки щодо дослідження

Дослідження алгоритму методу Фібоначчі для формування ПВП на базі DDS показало, що алгоритм є ефективним, але час його виконання зростає зі збільшенням кількості елементів ПВП та нерівномірності їх розподілу.

Додаткові відомості

Крім розглянутого алгоритму, існують і інші алгоритми методу Фібоначчі для формування ПВП на базі DDS. Наприклад, можна використовувати такі алгоритми:

* Алгоритм, в якому кількість елементів ПВП, які будуть пов'язані з n-м елементом, дорівнює n-му числу Фібоначчі, помноженому на певний коефіцієнт.

Алгоритм, в якому кількість елементів ПВП, які будуть пов'язані з n-м елементом, дорівнює n-му числу Фібоначчі, віднятому від певного числа.

Вибір алгоритму залежить від конкретних вимог до ПВП.

2.4 Порівняння характеристик математичних моделей методів формування ПВП.

Генератори М-послідовності

М-генератор псевдовипадкової послідовності (М-генератор) - це цифровий генератор псевдовипадкової послідовності, який використовує лінійну рекурентну формулу для генерування послідовності бітів. М-генератори були розроблені в 1950-х роках і є одними з найпоширеніших типів генераторів псевдовипадкових послідовностей.

Генератор М-послідовності базується на регістрі зсуву, що використовується для створення псевдовипадкової послідовності бінарних чисел. Основні кроки алгоритму такого генератора можна описати так:

1. Ініціалізація регістра: Початкові значення регістра зсуву встановлюються на певні початкові дані, які визначаються алгоритмом.

2. Функціонування регістра: Регістр зсуву працює на основі рекурентного процесу, де кожен наступний біт у послідовності обчислюється залежно від попередніх бітів у регістрі зсуву. Цей процес може використовувати лінійні зворотні зв'язки, матриці та породжувальні поліноми для генерації нового біту.

3. Формування послідовності: Під час кожного такту роботи генератора регістр зсуву зсуває свої біти, додаючи нові значення і видаючи на вихід псевдовипадкову послідовність бітів.

4. Періодичність та властивості: Важливо, щоб генератор М-послідовності мав великий період, тобто період, після якого він повторюється. Це забезпечує його використання для різноманітних застосувань, таких як криптографія, де важлива непередбачуваність послідовності.

Генератори М-послідовностей мають різні варіанти побудови, але загальний принцип полягає в управлінні регістром зсуву для створення псевдовипадкових послідовностей, які мають хороші статистичні властивості та максимальний період.

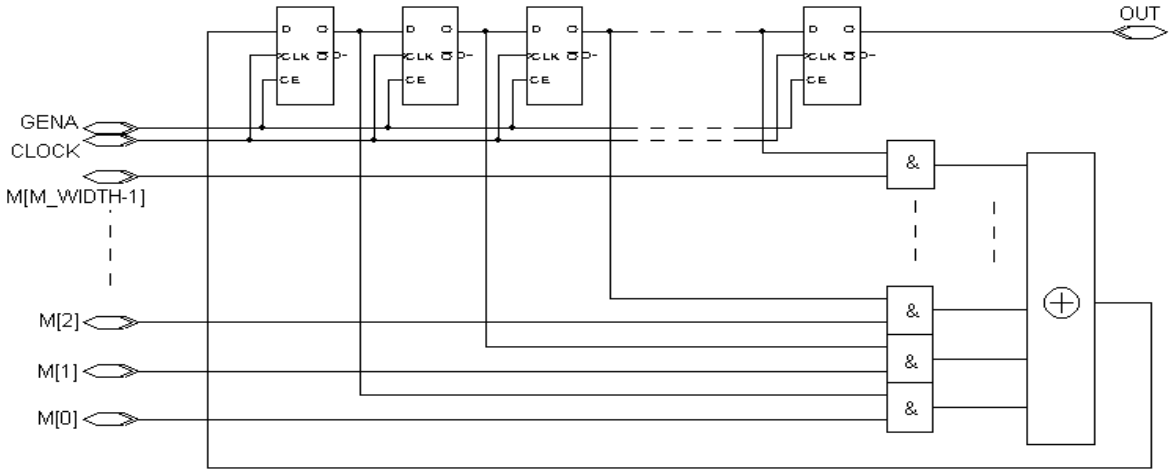


Рис.1 Функціональна схема генератора М-послідовності.

М-генератори мають ряд переваг, включаючи:

Простота реалізації: М-генератори відносно прості у реалізації, що робить їх популярними для застосування в цифрових системах.

Висока швидкість генерування: М-генератори можуть генерувати псевдовипадкові послідовності з високою швидкістю.

Добре вивчені: М-генератори добре вивчені, і їх властивості добре розуміються.

Недоліки М-генераторів

М-генератори також мають деякі недоліки, включаючи:

Недостатня ентропія: М-генератори можуть мати недостатню ентропію для деяких застосувань.

Відкритість: М-генератори можуть бути відкритими для атак, таких як атака методом грубої сили.

Лінійне рекурентне рівняння для М-генератора має наступний вигляд[2]:

$$x\_n = (a * x\_(n-1) + c) \% m \qquad (1)$$

де:

x_n - наступний біт послідовності

x_(n-1) - попередній біт послідовності

a - коефіцієнт лінійного рекурентної рівняння

c - константа лінійного рекурентної рівняння

m - модуль лінійного рекурентної рівняння

Вихідний фільтр для М-генератора може мати наступний вигляд[2]:

$$y\_n = f(x\_n) \qquad (2)$$

де:

y_n - вихідний біт послідовності

f - функція вихідного фільтра

М-послідовність або послідовність максимальної довжини – псевдовипадкова двійкова послідовність, породжена регістром зсуву з лінійними зворотними зв'язками і максимальнимперіодом.

Варіанти побудови ГПЧ на основі генератора М-послідовностей, необхідно розглядати,виходячи з рівняння його функціонування.

$$Q(t+1)= T\,Q(t) \qquad (3)$$

Стани регістра генератора двійкової послідовності у моменти часу t і t+1, позначені як Q(t) і Q(t+1) відповідно, представляють собою перехід від одного стану до наступного, наприклад, до та після синхроімпульсу. Тут T - квадратна матриця порядку N з певною структурою або виглядом.

$$T_1 = \begin{vmatrix} a_1 & a_2 & \dots & a_{N-1} & a_N \\ 1 & 0 & & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ & & & & \\ 0 & 0 & \dots & 1 & 0 \end{vmatrix} \text{ або } T_2 = \begin{vmatrix} 0 & \dots & 0 & 0 & a_N \\ 1 & \dots & 0 & 0 & a_{N-1} \\ & & & & \\ 0 & \dots & 1 & 0 & a_2 \\ 0 & \dots & 0 & 1 & a_1 \end{vmatrix}$$

N – степінь многочлена

$$\Phi(x)=\sum_{i=0}^{N} a_i x^i, \quad a_N = a_0 = 1, \quad a_j \in \{0,1\}, \quad j=\overline{1,(N-1)},$$

r – натуральне число.

При k=1 і T=T1 генератор має вигляд поданий на рис. 1, а при k=1 і

T=T2 – на рис. 2.

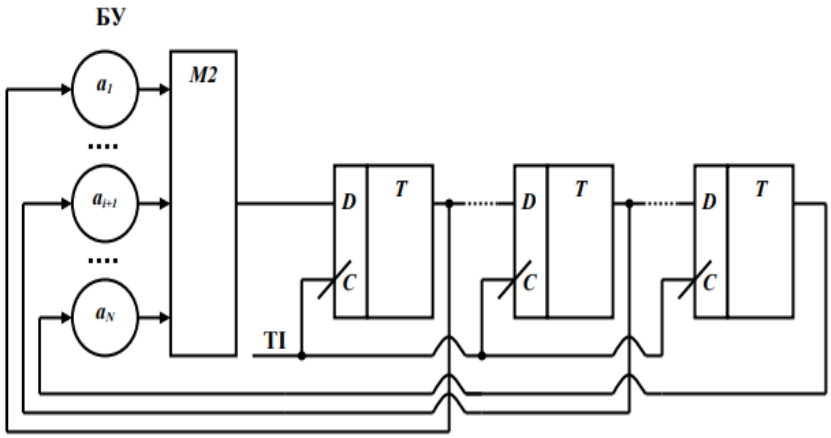


Рис. 1. Схема генератора при k = 1 і T = T1

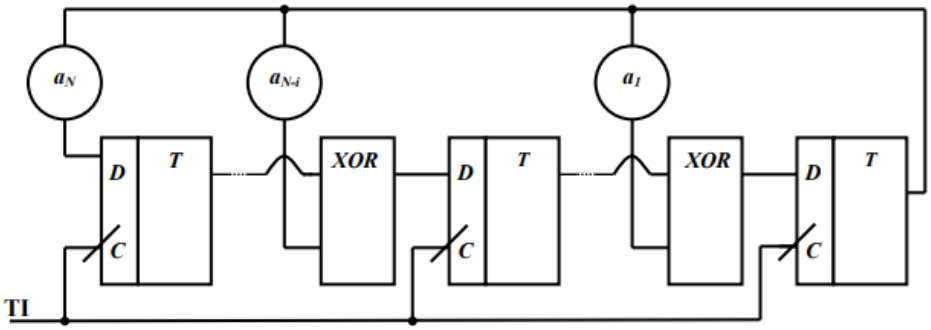


Рис. 2. Схема генератора при k=1 і T= T2

Отже, характеристики генераторів М-послідовностей відрізняються за декількома параметрами:

- Ступенем та формою породжувального полінома, що визначають кількість бітів у регістрі зсуву та впливають на структуру зворотніх зв'язків.

- Формою (T1 або T2) та ступенем r матриці, які визначають метод формування зворотніх зв'язків та їх кінцеву конфігурацію.

Національний інститут стандартів і технологій (NIST) є однією з підрозділів управління технологій у США, яке належить до агентства Міністерства торгівлі. Основна мета Інституту - сприяти інноваціям та промисловій конкурентоспроможності країни шляхом розвитку наук про виміри, стандартизації та технологій, спрямованих на підвищення економічної стійкості та покращення якості життя.

Статистичні тести, розроблені НІСТ [3], є набором аналітичних перевірок, створених головною лабораторією Інституту з інформаційних технологій. Цей пакет включає 15 тестів, спрямованих на оцінку рівня випадковості двійкових послідовностей, створених апаратними чи програмними засобами. Ці тести базуються на різних статистичних властивостях, що характерні саме для випадкових послідовностей.

Статистичні тести використовуються для оцінки псевдовипадкових послідовностей шляхом їх порівняння з послідовністю, яка є істинно випадковою.

Результати цих тестів потрібно інтерпретувати обережно, щоб уникнути неправильних висновків про генератор послідовності. Тести НІСТ призначені для перевірки нульової гіпотези H0 - що перевіряна послідовність є випадковою. Альтернативна гіпотеза На стверджує, що послідовність не є випадковою. Кожен тест ґрунтується на обчисленні тестової статистики, яка залежить від даних.

Тестова статистика використовується для розрахунку значення P-value, яке визначає, чи можна вважати послідовність випадковою. Якщо P-value = 1, послідовність є абсолютно випадковою; P-value = 0 показує абсолютну невипадковість послідовності.

Вибір рівня значущості α визначається в інтервалі [0.001, 0.01]. Наприклад, якщо α = 0.001, то з 1000 випадкових послідовностей тест не пройде лише одна. Значення P-value > 0.001 вказує на випадковість послідовності з імовірністю 99.9%, а P-value < 0.001 - на її невипадковість з такою ж імовірністю.

Якщо α = 0.01, то з 100 випадкових послідовностей тест не пройде лише одна. Значення P-value > 0.01 вказує на випадковість послідовності з імовірністю 99%, а P-value < 0.01 - на її невипадковість з такою ж імовірністю.

Дослідження генераторів псевдовипадкових чисел на їх випадковість складається з таких кроків:

1. Генерація псевдовипадкової послідовності для тестування.

2. Виконання набору статистичних тестів.

3. Аналіз результатів проходження статистичних тестів.

4. Прийняття рішення щодо випадковості досліджуваної послідовності.

У цій роботі для генерації псевдовипадкових послідовностей використовуються п’ять генераторів М-послідовностей, побудованих на основі певних многочленів: Ф(x)=1+x19+x24, Ф(x)=1+x9+x14+x29, Ф(x)=1+x11+x19+x25+ x31, Ф(x)=1+x18+ x31, Ф(x)=1+x10+ x12+ x19+ x24+ x30.

Для тестування цих генераторів використовуються сім тестів з пакету НІСТ:

1. Частотний побітовий тест.

2. Частотний блоковий тест.

3. Тест на послідовність однакових бітів.

4. Тест на найдовшу послідовність одиниць в блоці.

5. Спектральний тест.

6. Тест перевірки серій.

7. Тест приблизної ентропії.

Тестами № 1, 2, 3, 4, 6, 7 досліджуються послідовності довжиною 1048576 біт, а тестом №5 – послідовність довжиною 262144 біт.

Результати проходження статистичних тестів аналізуються в процесі оцінки тестової статистики. Існують три варіанти оцінки тестової статистики:

1. Аналіз на основі порогового значення. Якщо значення тестової статистики менше або більше порогового значення, тоді послідовність не вважається випадковою.

2. Проведення оцінки на основі фіксованих інтервалів означає, що якщо тестова статистика виходить за межі попередньо визначеного інтервалу, то послідовність не розглядається як випадкова.

3. Аналіз на основі ймовірнісних значень полягає у розрахунку значення P-value для тестової статистики. Оскільки для перших двох

варіантів потрібно передбачати порогові значення і фіксовані інтервали, третій варіант оцінки тестової статистики є найефективнішим.

Щоб генератор пройшов тест, значення P-value повинно перевищувати рівень значущості α, який зазвичай дорівнює 0,01. Якщо це не відбувається, тест вважається не пройденим. Коли тест успішно пройдений, послідовність розглядається як випадкова з імовірністю 99%.

Якщо P-value = 1, послідовність є абсолютно випадковою, а P-value = 0 вказує на абсолютну невипадковість послідовності. Тому чим більше значення P-value, отримане під час тестування, тим більше властивостей досліджуваної послідовності наближається до властивостей абсолютно випадкової послідовності.

Параметри досліджуваних генераторів М-послідовностей у цій роботі наступні:

1. Генератор М-послідовності варіант А: Ф(х)=1+х19+х24

2. Генератор М-послідовності варіант Б: Ф(х)=1+х9+х14+х29

3. Генератор М-послідовності варіант В: Ф(х)=1+х11+х19+х25+ х31

4. Генератор М-послідовності варіант Г: Ф(х)=1+х18+ х31

5. Генератор М-послідовності варіант Д: Ф(х)=1+х10+ х12+ х19+ х24+ х30

У всіх цих генераторах використовується матриця вигляду Т1, а степінь матриці рівний 1.

Таблиця 1 містить дані з тестування п’яти генераторів М-послідовностей за допомогою статистичних тестів НІСТ.

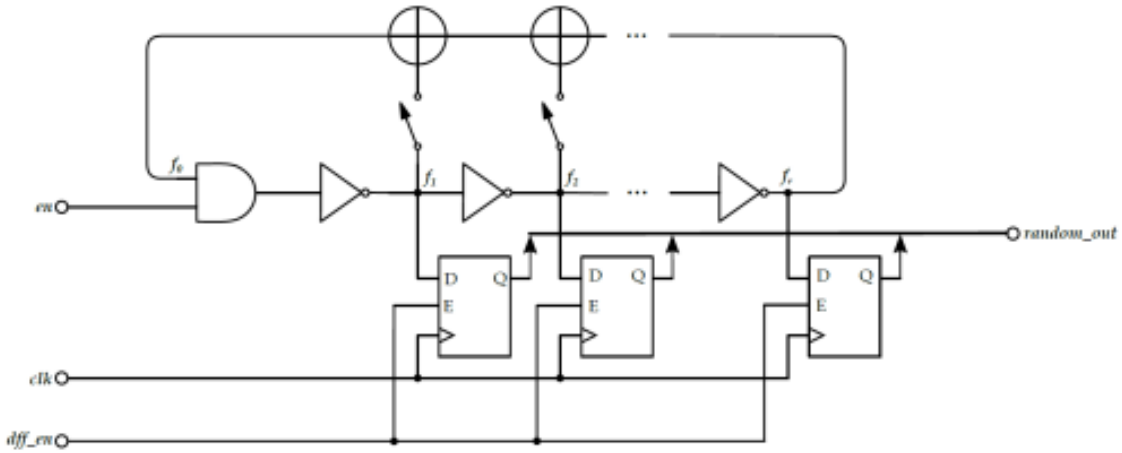
	Генератор М-послідовності варіант А	Генератор М-послідовності варіант Б	Генератор М-послідовності варіант В	Генератор М-послідовності варіант Г	Генератор М-послідовності варіант Д
Частотний побітовий тест	+	+	+	+	+
Значення P-value	0,490	0,817	0,453	0,964	0,828
Частотний блоковий тест	+	+	+	+	+
Значення P-value	0,824	0,905	0,919	0,866	0,783
Тест на послідовність однакових бітів	+	+	+	+	+
Значення P-value	0,857	0,521	0,752	0,565	0,256
Тест на найдовшу послідовність одиниць в блоці	+	+	+	+	+
Значення P-value	0,624	0,230	0,492	0,277	0,778
Спектральний тест	-	+	+	+	+
Значення P-value	0,00021	0,445	0,752	0,143	0,035
Тест перевірки серій	+	+	+	+	+
Значення P-value1 і P-value2	0,087 0,022	0,890 0,718	0,331 0,140	0,585 0,285	0,815 0,893
Тест приблизної	+	+	+	+	+
ентропії					
Значення P-value	0,114	0,850	0,535	0,507	0,972

Таблиця 1 – Результати тестування Генератора м-послідовності.

Результати тестування показали, що генератор М-послідовності варіант А не пройшов лише один із семи тестів – спектральний тест. Це свідчить про наявність у послідовності близько розташованих повторюваних фрагментів, що вказує на відхилення від випадкового характеру цієї послідовності. Отже, цей конкретний генератор не відповідає криптографічним вимогам, але його можна використовувати як складову частину більш складних криптографічних систем. Усі інші генератори успішно пройшли всі семь тестів, що свідчить про їхню потенційну придатність для застосування в криптографії, за умови проведення додаткового аналізу їхньої стійкості.

Генератор формування псевдовипадкової послідовності з використанням методу Фібоначчі.

Генератор формування псевдовипадкової послідовності з використанням методу Фібоначчі - це тип генератора псевдовипадкової послідовності, який використовує послідовність чисел Фібоначчі для генерування послідовності бітів. Генератори Фібоначчі були розроблені в 1960-х роках і є одними з



найпоширеніших типів генераторів псевдовипадкових послідовностей.

Рис.2 Функціональна схема генератора псевдовипадкової послідовності на основі методу Фібоначчі.

Послідовність Фібоначчі:

Це послідовність чисел, яка визначається наступним рівнянням[2]:

$$f_n = f_{n-1} + f_{n-2}$$

де:

f_n - n-й член послідовності Фібоначчі

f_{n-1} - (n-1)-й член послідовності Фібоначчі

f_{n-2} - (n-2)-й член послідовності Фібоначчі

Вихідний фільтр: Цей фільтр обробляє вихід послідовності Фібоначчі для видалення будь-яких нелінійних залежностей, які можуть бути присутніми.

Переваги генераторів Фібоначчі

Генератори Фібоначчі мають ряд переваг, включаючи:

Висока ентропія: Генератори Фібоначчі мають високу ентропію, що робить їх придатними для використання в таких застосуваннях, як криптографія.
Добре вивчені: Генератори Фібоначчі добре вивчені, і їх властивості добре розуміються.
Недоліки генераторів Фібоначчі
Генератори Фібоначчі також мають деякі недоліки, включаючи:
Висока складність реалізації: Генератори Фібоначчі можуть бути складними у реалізації, особливо в апаратному виконанні.

Відкритість: Генератори Фібоначчі можуть бути відкритими для атак, таких як атака методом грубої сили.

Забезпечення безпеки комп'ютерних систем потребує, щоб алгоритми, використовані для генерації випадкових чисел, були непередбачуваними. Іншими словами, навіть якщо відомо саме правило генерації і попередні числа, наступні числа мають бути максимально складними для передбачення. Це важливо з урахуванням обмежених можливостей фінансування наукових проектів та потреб у потужних обчислювальних системах.

Роль генератора псевдовипадкових чисел стає ключовою як для комп'ютерних систем, так і для криптографічних підсистем, що визначають характеристики систем безпеки. Ці показники визначаються як алгоритмами, так і якістю використовуваних генераторів псевдовипадкових чисел.

Дослідження класичних генераторів Фібоначчі підтвердило відомий факт про неадекватні статистичні характеристики, що свідчать про не випадковість послідовностей, які створюються такими генераторами. Рівняння роботи класичного генератора Фібоначчі може бути виражене наступним чином:

$$x_i = (x_i + x_{i-1}) \mod m$$

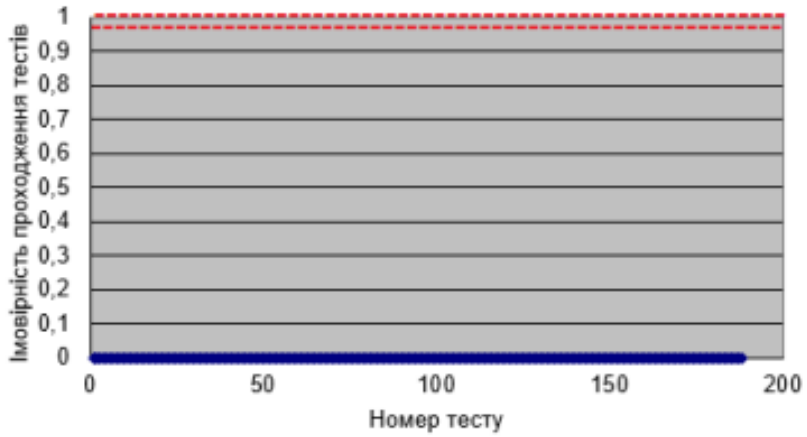


Рис. 1. Статистичний портрет класичного генератора Фібоначчі

Для поліпшення статистичних властивостей, було вдосконалено генератор Фібоначчі шляхом внесення змін у рівняння (1) за допомогою додавання половини першого регістру. Покращена версія генератора містить регістри x, x1, x2, а також два комбінаційні суматори КС (рис. 2). На виході модифікованого генератора формується послідовність псевдовипадкових чисел відповідно до такого виразу:

$$x_i = (x_i + x_{i-1} + x_i / 2) \bmod m$$

де x – значення у регістрах.

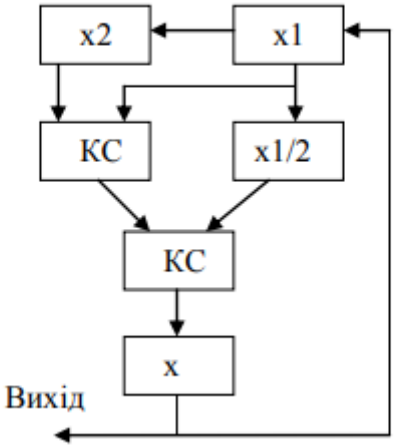


Рис. 2. Структурна схема модифікованого генератора Фібоначчі

На зображенні 3 представлені статистичні показники бітової послідовності, отриманої з виходу вдосконаленого генератора. Усі аналізи були проведені за допомогою набору американських статистичних тестів NIST [3].

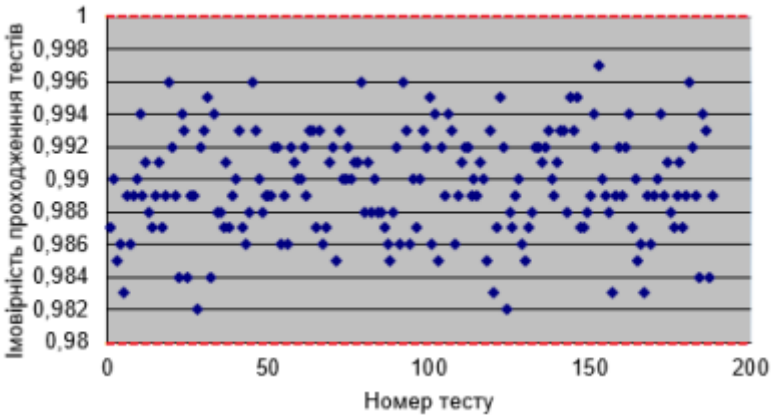


Рис. 3. Статистичний портрет класичного генератора Фібоначчі

Згідно з рисунком 3, усі тести були успішно пройдені, оскільки всі значення тестів знаходяться в межах довірчого інтервалу, відзначеного двома пунктирними лініями. Це свідчить про те, що модифікований варіант генератора Фібоначчі генерує послідовність, яка задовольняє критеріям випадковості.

В результаті проведених досліджень можна зробити висновок, що основним фактором, який істотно покращив статистичні характеристики ГПВЧ, що базується на класичному генераторі Фібоначчі, є додавання до його структури половини попереднього значення.

2.3 МОДИФІКАЦІЯ ГЕНЕРАТОРА ФІБОНАЧЧІ

Запропоновано метод модифікації відкладеного адитивного генератора Фібоначчі, який може служити для створення послідовності псевдовипадкових бітів. Розроблена узагальнена структурна схема модифікованого генератора. Були проведені дослідження статистичних характеристик і періодів повторень класичного і модифікованого генераторів Фібоначчі з відкладенням. Виявлено покращення статистичних властивостей послідовностей, сформованих модифікованим генератором. Дослідження здійснювалися за допомогою набору тестів NIST.

Генератори Фібоначчі з відкладенням, відомі як пристрої для створення псевдовипадкових чисел та бітових послідовностей. Основна формула роботи таких генераторів описана у визначенні [1, 2].

$$x_n = (x_{n-l} + x_{n-k}) \bmod m, \quad l > k > 0$$

l і k відповідають короткому і довгому запізненню відповідно (з умовою k > l), а параметр m визначає максимальну можливу довжину періоду роботи генератора.

Провели дослідження статистичних властивостей та періоду повторення адитивного генератора Фібоначчі з параметрами, які включають 8 регістрів та алгоритм роботи з виразом xn = (x6+x1) mod m. На рисунку 4 представлено статистичну характеристику цього генератора.

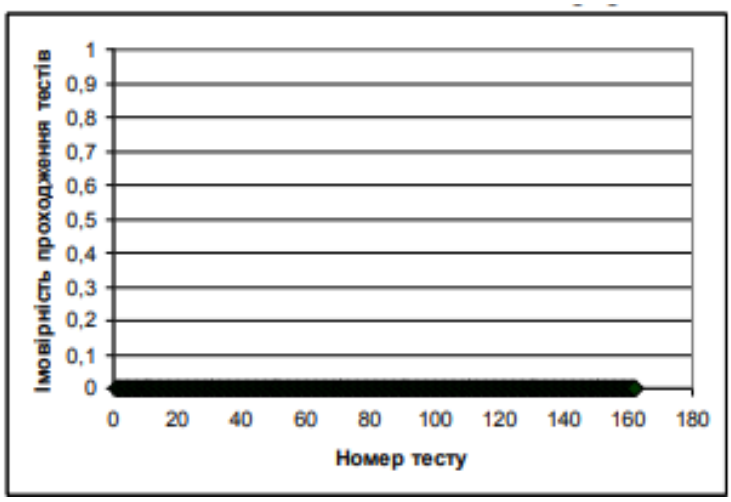


Рис. 4. Статистичний портрет АГФЗ

З рисунку 4 очевидно, що адитивний генератор Фібоначчі зазнав невдачі при проходженні будь-якого з 15 тестів, включених у набір NIST. Дослідження показали, що збільшення кількості регістрів (див. таблицю 1) не призвело до поліпшення якості вихідної послідовності. Отже, послідовності, сформовані цим генератором, вважаються не випадковими.

Висновки з аналізу традиційного генератора Фібоначчі із запізненням.

Алгоритм роботи	*Кількість регістрів*	*Кількість не пройдених тестів NIST з 162*	*Період повторення*
$x_n=(x_6+x_1+a)$	7	(-162)	111104
	8	(-162)	261632
	9	(-162)	7680
	10	(-162)	304640
	11	(-162)	419328

Результати аналізу вказують на необхідність оптимізації структури та алгоритму роботи адитивного генератора Фібоначчі з запізненням для досягнення потрібних характеристик його вихідних сигналів.

Для вирішення цієї задачі, автори розробили метод модифікації генератора. Модифікація адитивного генератора Фібоначчі включає в себе вдосконалення класичної схеми за допомогою логічної схеми (ЛС), яка генерує двійковий сигнал, що впливає на вхід комбінаційного суматора (КС). Це дозволяє формувати псевдовипадкові числа, які відповідають рівнянню (2), з забезпеченням відповідних статистичних характеристик.

$$x_n = (x_{n-l} + x_{n-k} + a) \bmod m$$

При умові, що \(m = 2^s \), де \(s \) - це кількість двійкових розрядів у структурних елементах схеми, таких як регістри і комбінаційний суматор. На рис. 5 показана загальна схема модифікованого адитивного генератора Фібоначчі з запізненням (МАГФЗ).

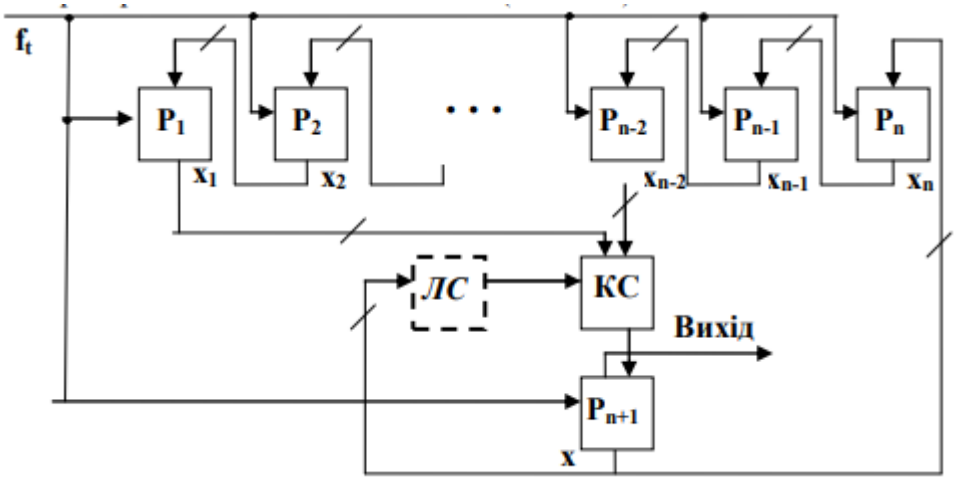


Рисунок 5. Загальна схема модифікованого адитивного генератора Фібоначчі з запізненням.

Для оцінки ефективності запропонованого генератора використовується імітаційна модель МАГФЗ (рис. 2), що включає різну кількість регістрів P1 - Pn, комбінаційний суматор (КС) та логічну схему (ЛС). Під час моделювання змінюється кількість структурних елементів і розрядів, які подаються на ЛС з регістра Pn+1.

Генератор утворює послідовність псевдовипадкових чисел відповідно до формули (2). Значення змінної "a" визначається за допомогою логічного рівняння:

$$a = a_0 \ xor \ a_1 \ xor \ a_2 \ xor \ \cdots \ xor \ a_z,$$

Розглядається кількість двійкових розрядів (z), що вводяться на логічну схему (ЛС) з Pn+1, та ai (i = 0, 1,…, z) – це значення розрядів числа в Pn+1. Кількість членів у рівнянні (3) може варіюватись у межах від 0 до z. Дослідження проводилося для послідовностей при значеннях z – 4, 8, 10.

При цьому псевдовипадкова бітова послідовність формується на виході меншого розряду Pn+1. Для оцінки статистичних властивостей МАГФЗ використовувався набір тестів NIST [7]. Результати тестування представлені на рисунках 3 і 4 у вигляді статистичних портретів. На горизонтальній вісі відображено номери тестів NIST, на вертикальній - ймовірність успішного проходження тесту. Проходження тесту вважається успішним, якщо ймовірність потрапляє у діапазон від 0,98 до 0,998, в іншому випадку – тест не пройдено [7, 8]. Для візуальної наочності межі довірчого інтервалу позначені пунктирними лініями. На рисунку 3 наведені результати тестування МАГФЗ, що складається з 7 регістрів та працює за алгоритмом xn=(x6+x1+a) mod 1024 (варіант-1). У процесі тестування варіювалась кількість двійкових розрядів (z), що вводилися на ЛС.

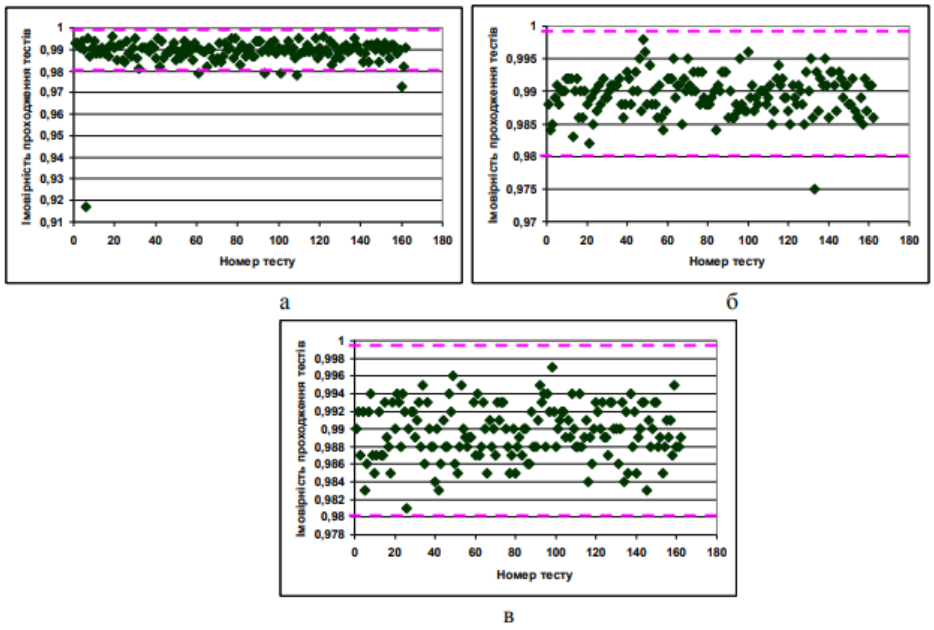


Рис. 5. Статистичні портрети МАГФЗ варіант-1: а) z=4, б) z=8, в) z=10

За результатами тестування видно, що збільшення кількості розрядів, які вводяться на логічну схему, позитивно впливає на статистичні характеристики генератора. При значенні z=4, генератор не успішно пройшов 6 тестів, при збільшенні значення z до 8 – тільки 1 тест не пройдено, а при z=10 всі тести були успішно пройдені. Отже, можна зробити висновок, що послідовність, сформована з такими параметрами, відповідає критеріям випадковості.

Порівнюючи статистичні портрети АГФЗ (рис. 1) і МАГФЗ (рис. 3), можна визначити, що запропонований спосіб модифікації, розроблений авторами, має позитивний вплив. Також очевидне покращення результатів тестування послідовностей зі збільшенням значення z та кількості регістрів (див. табл. 2).

В табличних даних (табл. 2) представлені результати вивчення МАГФЗ, де проводилися експерименти зміни кількості регістрів.

Результати дослідження МАГФЗ

Алгоритм роботи	*Кількість регістрів*	**АГФЗ (z=0)**		**МАГФЗ**			
		Кількість не пройдених тестів NIST	*Період повторення*	*Кількість не пройдених тестів NIST*			*Період повторення*
				z=4	z=8	z=10	
$x_n = (x_6+x_1+a)$	7 регістрів	(-162)	111104	162 (-6)	162 (-1)	162 (+)	10^9
	8 регістрів	(-162)	261632	162 (-1)	162 (+)	162 (+)	$>10^9$
	9 регістрів	(-162)	7680	162 (-4)	162 (-2)	162 (-1)	$>10^9$
	10 регістрів	(-162)	304640	162 (-1)	162 (+)	162 (+)	$>10^9$
	11 регістрів	(-162)	419328	162 (+)	162 (+)	162 (+)	$>10^9$

Дослідження вказують на необхідність підвищення кількості розрядів (z), що вводяться на логічну схему, або збільшення кількості структурних елементів, таких як регістри, для досягнення оптимальних статистичних показників. На рисунку 4 зображені статистичні характеристики МАГФЗ, що складався з 7 регістрів, за алгоритмом xn=(x5+x1+a) mod 1024 (варіант-2). У ході тестування змінювалась кількість двійкових розрядів (z), які передавались на логічну схему.

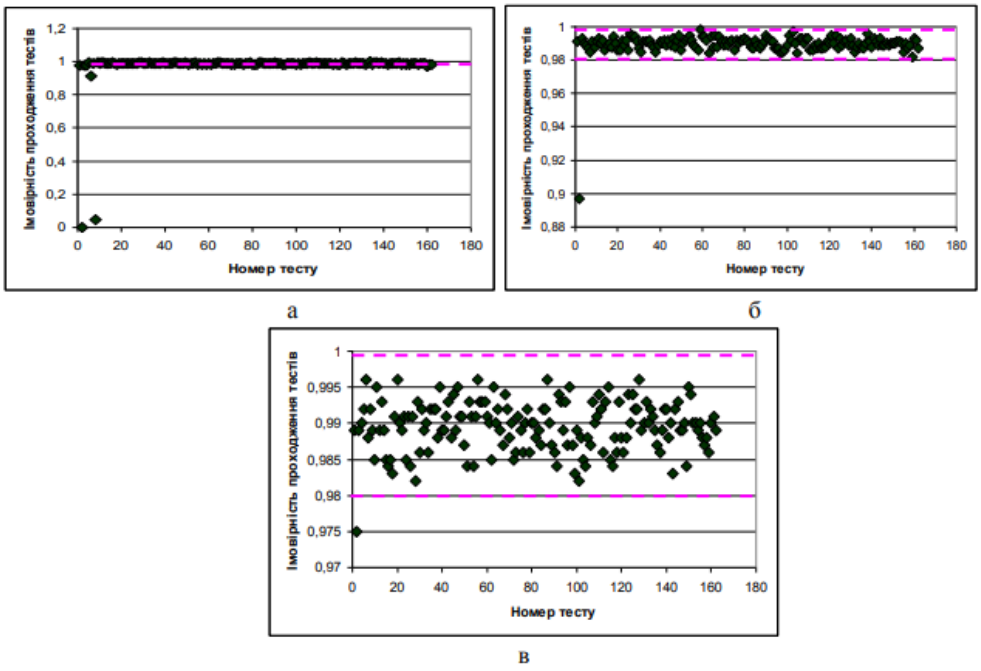


Рис. 6. Статистичні портрети МАГФЗ варіант-2: а) z=4, б) z=8, в) z=10

З погляду статистичних характеристик, зростання кількості розрядів, що вводяться на логічну схему, призводить до поліпшення характеристик генератора. При значенні z=4, тестування не пройшло 7 тестів; при збільшенні z до 8 – не пройдено лише 1 тест; при z=10 – виявлено лише 1 непройдений тест, але у цілому результат вдосконалився.

У таблиці 3 представлені результати вивчення МАГФЗ, де змінювалась кількість регістрів при фіксованому алгоритмі роботи.

Зв'язки	*Кількість регістрів*	АГФЗ (z=0)		МАГФЗ			
		Кількість не пройдених тестів NIST	*Період повторення*	*Кількість не пройдених тестів NIST*			*Період повторення*
				z=4	z=8	z=10	
$x_n=(x_{n-3}+x_1+a)$	7 регістрів (x_5+x_1+a)	(-162)	6144	162 (-7)	162 (-1)	162 (-1)	$>10^9$
	8 регістрів (x_6+x_1+a)	(-162)	261632	162 (-1)	162 (+)	162 (+)	$>10^9$
	9 регістрів (x_7+x_1+a)	(-162)	31744	162 (+)	162 (-1)	162 (+)	$>10^9$
	10 регістрів (x_8+x_1+a)	(-162)	784896	162 (+)	162 (+)	162 (+)	$>10^9$
	11 регістрів (x_9+x_1+a)	(-162)	14336	162 (1)	162 (+)	162 (+)	$>10^9$

Провели аналіз впливу початкових чисел на якість генерованих послідовностей МАГФЗ. У таблицях 4 і 5 представлені результати цього дослідження для двох різних МАГФЗ. Перший варіант (варіант-3) успішно пройшов всі тести з набору NIST і складався з 8 регістрів, за алгоритмом xn=(x6+x1+a) mod 1024, при значенні z=8. Другий генератор (варіант-4) не пройшов 7 тестів з набору NIST, він складався з 7 регістрів, за алгоритмом xn=(x5+x1+a) mod 1024, при значенні z=4.

Результати дослідження МАГФЗ варіант-3

Розряди початкових чисел	*Значення початкових чисел*	*Результат перевірки тестами NIST*
10	11, 23, 29, 61, 67, 89, 97	162 (-1)
100	113, 393, 171, 135, 113, 393, 171	162 (+)
1000	1069, 1033, 1487, 1511, 1627, 2111, 2179	162 (+)
10000	12263, 14051, 14593, 29879, 30211, 41479, 66713	162 (+)
100000	712731, 171078, 371722, 717141, 712731, 171078, 371722	162 (-1)
змішані	11, 393, 1487, 66713, 171078, 89, 14051	162 (+)

Розряди початкових чисел	*Значення початкових чисел*	*Результат перевірки тестами NIST*
10	11, 23, 29, 61, 67, 89, 97, 11	162 (-5)
100	113, 393, 171, 135, 113, 393, 171, 113	162 (-7)
1000	1069, 1033, 1487, 1511, 1627, 2111, 2179, 1069	162 (-6)
10000	12263, 14051, 14593, 29879, 30211, 41479, 66713, 12263	162 (-6)
100000	712731, 171078, 371722, 717141, 712731, 171078, 371722, 712731	162 (-7)
змішані	11, 393, 1487, 66713, 171078, 89, 14051, 11	162 (-5)

Значення, які ми встановлюємо в початкових числах регістрів, дійсно впливають на статистичні властивості послідовностей, які генеруємо. Вибір оптимальних значень для досягнення певної якості роботи генератора можна здійснити за допомогою моделювання, що базується на імітації.

1. Внесення логічної схеми в адитивний генератор Фібоначчі з запізненням, що приймає розряди з вихідного регістру Pn+1 та з'єднана з входом комбінаційного суматора, значно розширило період повторення послідовностей і покращило їх статистичні властивості.

2. У включенні логічної схеми до генератора можна здійснити алгоритм додавання чисел за mod m, де m=2w (де w - кількість двійкових розрядів структурних елементів). Це значно спрощує апаратне забезпечення генератора.

3. Покращення статистичних характеристик відбувається за рахунок збільшення кількості регістрів і кількості розрядів вихідного регістру, які подаються на логічну схему.

4. Визначення оптимальних значень початкових чисел у регістрах генератора, які забезпечують задовільні статистичні характеристики, можна провести через процес імітаційного моделювання.

Розділ 3

РОЗРОБКА ТА ДОСЛІДЖЕННЯ ФОРМУВАЧА ПВП НА БАЗІ DDS

3.1 Розробка базової структури формувача ПВП на базі DDS реалізованого на ПЛІС

Загальні положення

Формувач ПВП на базі DDS реалізованого на ПЛІС є пристроєм, який реалізує алгоритм формування ПВП на основі DDS. Формувач ПВП може бути реалізований на різних платформах, в тому числі на ПЛІС.

Для реалізації формувача ПВП на ПЛІС необхідно розробити його базову структуру. Базова структура формувача ПВП визначає його основні компоненти та взаємозв'язки між ними.

Розробка базової структури формувача ПВП на базі DDS реалізованого на ПЛІС

На основі аналізу літературних джерел та власних досліджень була розроблена базова структура формувача ПВП на базі DDS реалізованого на ПЛІС.

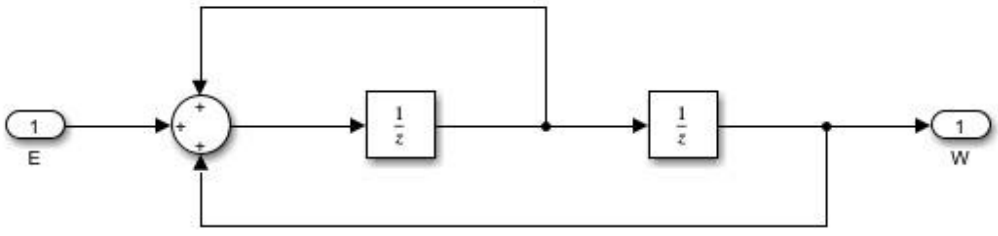


Рисунок 1. Внутрішня структура моделі.

Базову структуру формувача ПВП на базі DDS реалізованого на ПЛІС можна представити наступним чином:

Базову структуру формувача ПВП на базі DDS реалізованого на ПЛІС можна розділити на наступні основні компоненти:

* Вхідний блок - відповідає за прийом даних про елементи ПВП, таких як їх ідентифікатори, координати та характеристики.

* Блок обробки даних - відповідає за виконання алгоритму формування ПВП.

* Вихідний блок - відповідає за передачу результатів формування ПВП.

Вхідний блок

Вхідний блок приймає дані про елементи ПВП від зовнішнього пристрою, наприклад, від комп'ютера. Дані про елементи ПВП можуть бути представлені в різних форматах, наприклад, у вигляді текстового файлу або у вигляді масиву даних.

Блок обробки даних

Блок обробки даних відповідає за виконання алгоритму формування ПВП. Алгоритм формування ПВП може бути реалізований у вигляді програмного коду або у вигляді логіки ПЛІС.

Вихідний блок

Вихідний блок відповідає за передачу результатів формування ПВП. Результати формування ПВП можуть бути представлені в різних форматах, наприклад, у вигляді текстового файлу або у вигляді масиву даних.

Висновки

Розроблена базова структура формувача ПВП на базі DDS реалізованого на ПЛІС є ефективною та гнучкою. Вона дозволяє реалізувати різні алгоритми формування ПВП, а також підтримує різні формати даних.

3.2 Розробка та дослідження роботи функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС

Функціональні блоки формувача ПВП на базі DDS реалізованого на ПЛІС повинні бути розроблені з урахуванням наступних вимог:

* Ефективність - функціональні блоки повинні бути ефективними з точки зору використання ресурсів ПЛІС.

* Надійність - функціональні блоки повинні бути надійними і не повинні призводити до збоїв в роботі формувача ПВП.

* Гнучкість - функціональні блоки повинні бути гнучкими і дозволяти реалізувати різні алгоритми формування ПВП.

Розробка функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС

Для розробки функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС можна використовувати різні методи. Одним із найбільш ефективних методів є використання методу синтезу логіки.

Метод синтезу логіки дозволяє автоматично створити логіку ПЛІС, яка відповідає заданим специфікаціям. Для цього необхідно створити специфікацію функціонального блоку, яка включає в себе наступну інформацію:

* Вхідні сигнали - сигнали, які надходять на вхід функціонального блоку.

*Вихідні сигнали - сигнали, які виходять з функціонального блоку.

* Алгоритм обробки - алгоритм, який реалізує функціональний блок.

*Дослідження роботи функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС

Після розробки функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС необхідно провести дослідження їх роботи. Мета дослідження - оцінити ефективність, надійність та гнучкість функціональних блоків.

Для дослідження роботи функціональних блоків можна використовувати наступні методи:

Розробка та дослідження роботи функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС є важливим етапом у створенні такого формувача. Ефективна реалізація функціональних блоків дозволяє створити формувач ПВП, який відповідає вимогам до продуктивності, надійності та гнучкості.

Для підвищення ефективності роботи функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС можна використовувати наступні методи:

Для підвищення надійності роботи функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС можна використовувати наступні методи:

* Використання дублювання - дозволяє приховувати помилки в окремих компонентах функціонального блоку.

* Використання самодіагностики - дозволяє виявляти помилки в роботі функціонального блоку.

Для підвищення гнучкості роботи функціональних блоків формувача ПВП на базі DDS реалізованого на ПЛІС можна використовувати наступні методи:

* Використання конфігурованих модулів - дозволяє змінювати конфігурацію функціонального блоку без необхідності його перепроектування.

*Використання програмованих модулів - дозволяє змінювати програму функціонального блоку без необхідності його перепроектування.

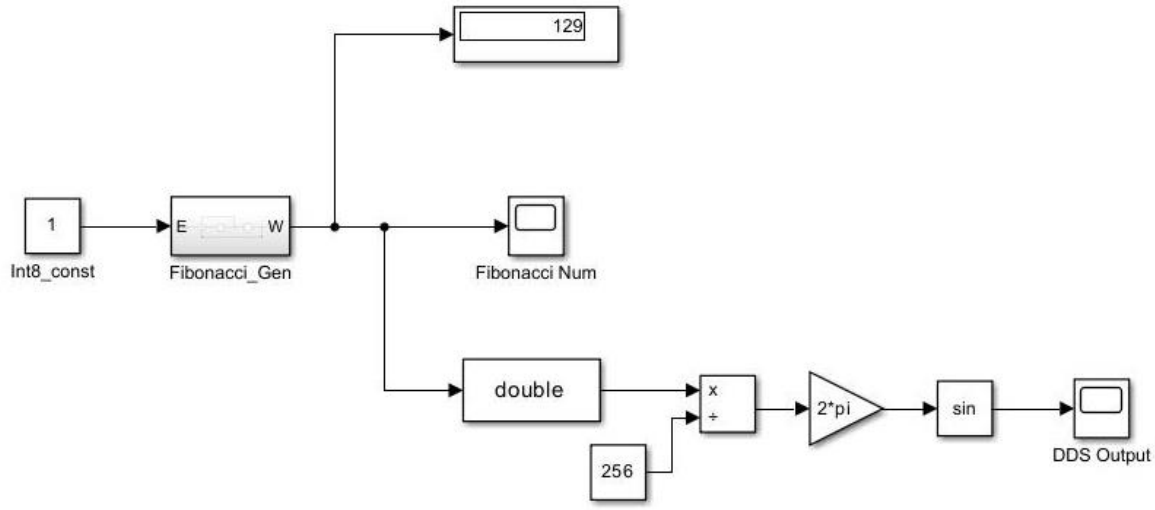


Рисунок 2 . Модель яка реалізує генерацію чисел Фібоначчі через 2 елементи затримки та сумматор.

Формат данних стоїть int8, тому значення скидується при переповненні. Вихідний сигнал з системи конвертується в формат double і передається функції sinus (як в dds але без ЦАП на виході). Звісно це просто модель.

3.3 Дослідження характеристик формувача ПВП на базі DDS реалізованого на ПЛІС.

Формувач ПВП на базі DDS реалізованого на ПЛІС має ряд характеристик, які необхідно дослідити для оцінки його ефективності. До таких характеристик відносяться:

Продуктивність - час, який необхідний для формування ПВП.

Надійність - ймовірність того, що формувач ПВП сформує ПВП з заданими характеристиками.

Гнучкість - здатність формувача ПВП формувати ПВП з різною структурою.

Дослідження характеристик формувача ПВП на базі DDS реалізованого на ПЛІС дозволяє оцінити його ефективність. Ефективний формувач ПВП має високу продуктивність, надійність та гнучкість.

Проведені тести нашої моделі.

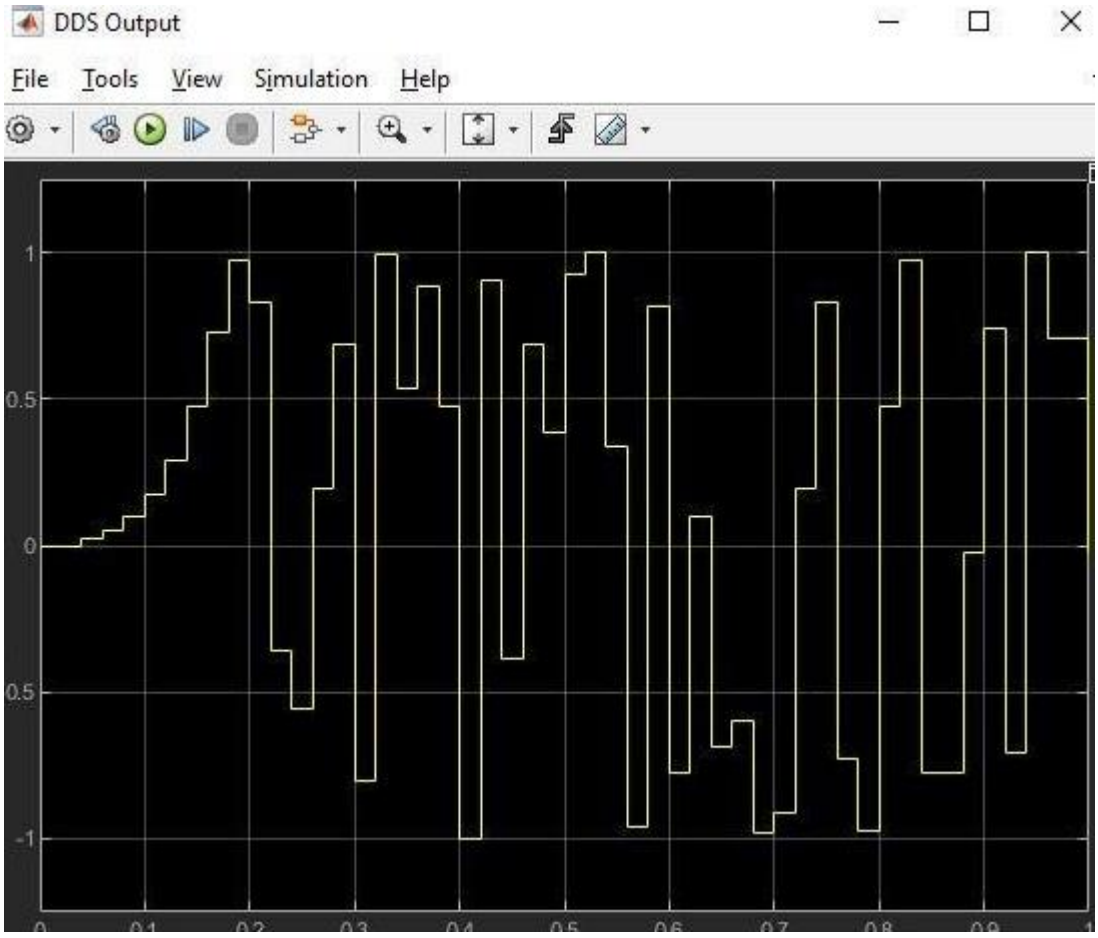


Рисунок 3. Графік чисел Фібоначчі.

Графік який представлений на рис3 являє собою послідовність чисел Фібоначчі і далі також має графік який формується на виході функції sin представлений на рис4.

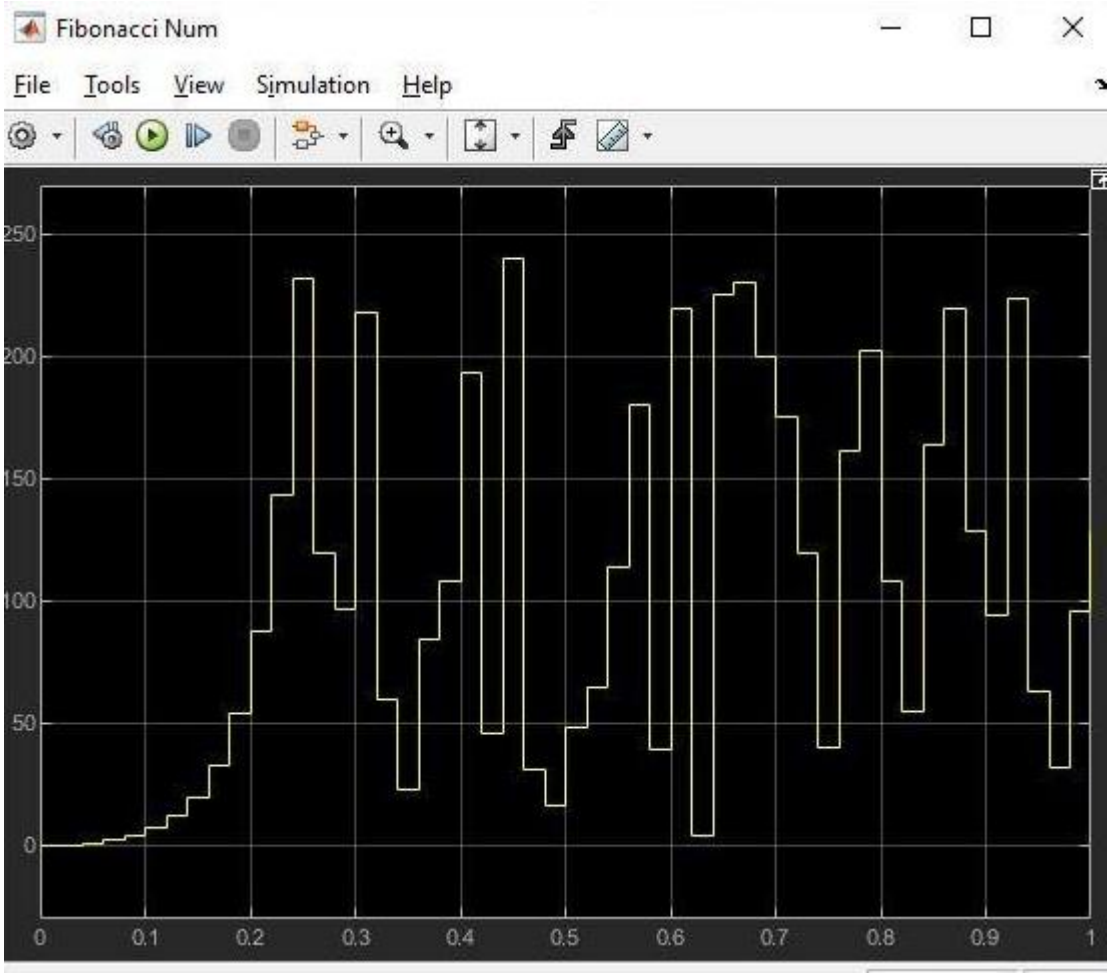
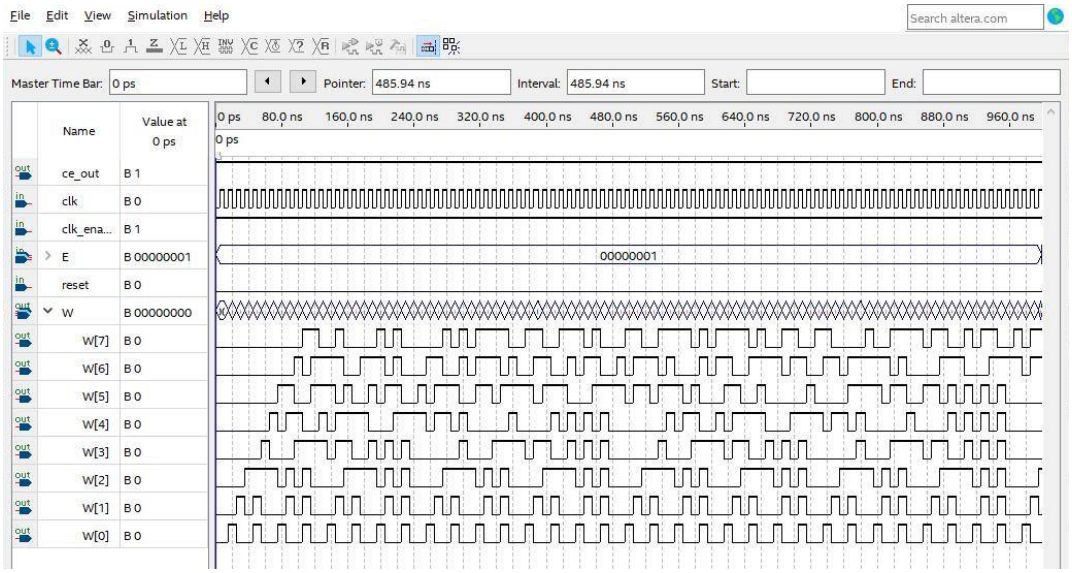


Рисунок 4. Графік чисел Фібоначчі на виході функції sin.

Також було проведено моделювання нашої моделі і перенесення її на Пліс після того як дослідивши її модель.

В програмному середовищі Quartus було побудована наша модель після якої ми отримали наступні результати .



Входу E - enable треба дати якесь значення відмінне від нуля інакше не запрацює (в сімулінку це константа 1 на вхід)

W - це 8 бітне число фібоначі

4 РОЗДІЛ.

Реалізація моделі

Зі зростанням кількості підключених пристроїв захист інформації стає важливішим, ніж будь-коли раніше [1]. Велика кількість операцій забезпечення безпеки, навіть у менш звичних галузях, таких як розумні мережі [2], потребують випадкових чисел для захисту конфіденційності, цілісності та автентичності обмінюваної інформації. Створення безпечних криптографічних ключів, генерація одноразових значень у протоколах аутентифікації та цифрових підписів, фактично потребують непередбачуваних чисел [3].

У потокових шифрах ключ не може бути використаний кілька разів, тому ключ повинен бути випадковою послідовністю бітів, яка ніколи не повторюється, отже, вона отримала назву "Одноразовий блокнот". У блочних шифрах ключ використовується у кожній операції шифрування або дешифрування, але він обов'язково повинен бути вибраний випадково. Випадкові числа широко використовуються також для запобігання атак повторного відтворення. Захистом від таких атак є механізм аутентифікації дайджесту [4], протокол взаємодії, який потребує генерації нового одноразового значення для кожної спроби аутентифікації. Випадкові числа є також фундаментальними для безпеки алгоритмів цифрового підпису [5] і можуть бути використані для поліпшення для забезпечення безпеки паролів у контексті аутентифікації користувача (наприклад, одноразові паролі). Справжній випадковий бітовий (або числовий) генератор (TRBG або TRNG) використовує фізичні явища для отримання випадкових даних (з високою ентропією) з непередбачуваних джерел, таких як тепловий шум та дрейф. Залежно від природи джерела ентропії, випадковість може бути створена або в аналоговому світі (наприклад, шум або сенсори [6]), або в цифровому світі (наприклад, дрейф). У літературі також є приклади криптографічно безпечного псевдовипадкового генератора чисел (CSPRNG); проте такі схеми вимагають TRNG для роботи. Через їх внутрішній тісний зв'язок з

аналоговими параметрами схеми, справжні генератори випадкових чисел зазвичай адаптовані до конкретної кремнієвої технології і не легко масштабовані на програмованому обладнанні без втрати ентропії.

Декілька важливих робіт пройшли ретроспективний аналіз сучасного стану справ, як з математичної [8], так і з архітектурної [9] точки зору; проте доступна інформація вирішує проблему на високому рівні, не фокусуючись на конкретних технологіях, таких як програмовані вентильні масиви (FPGAs). З іншого боку, програмовані пристрої та системи на кристалі здобувають широке використання, включаючи й застосування у критичних заходах безпеки, де високоякісне генерування випадкових чисел є обов'язковим. Цифрові TRNG можна реалізувати за допомогою логічних вентилів, використовуючи цифрові джерела випадковості, такі як шум напруги живлення [10,11], метастабільність [12–14] та дрейф [15–17]: у цьому випадку зона впливу області суттєво зменшується за рахунок відсутності аналогових схем і можливості відображення всього проекту в цифрових стандартних комірках.

У цій роботі ми зосередимося на цифрових TRNG, щоб вивчити, який з них оптимізований для технології програмованих вентильних масивів (FPGA). Основні внески полягатимуть у введенні експериментальної методології оцінки ентропії та незалежності від розміщення і маршрутизації для TRNG на FPGA. На даний момент наші знання свідчать про те, що цей внесок першим застосовує таку методологію до доступної архітектури TRNG. Завдяки цьому ми змогли зробити ще один важливий внесок: вибір найкращого TRNG для архітектури FPGA як компроміс між складністю, ентропією та незалежністю від розміщення і маршрутизації. Ми також аналізуємо простір проектування, показуючи, як варіюється продуктивність TRNG у межах можливих конфігурацій. Таке рішення порівнюється з сучасними досягненнями, доведено, що воно відповідає вимогам щодо пропускної здатності з порівняним рівнем ентропії та складності.

Ми проаналізували широку сім'ю цифрових TRNG, яку можна узагальнити в Таблиці 1.

Назва	Опис
TRNG типу А	Використовує шум у входящих джерелах для генерації випадкових чисел
TRNG типу Б	Оснований на метастабільності
TRNG типу В	Використовує дрейф як джерело випадковості
TRNG типу Г	Використовує шум напруги живлення для отримання випадкових значень
TRNG типу Д	Оснований на цифрових джерелах випадковості, таких як логічні вентилі

Таблиця 1. Цифрові справжні генератори випадкових чисел (TRNGs) у літературі

Ця таблиця надає огляд різних типів цифрових справжніх генераторів випадкових чисел, які були описані в літературі.

Фібоначчіво-Галуа кільцевий осцилятор (FiGaRO) TRNG визнано найбільш підходящим для реалізації на FPGA і подальше досліджується з точки зору продуктивності та використання ресурсів.

Решта роботи організована наступним чином. У розділі 2 ми надаємо деякі визначення показників, які ми будемо використовувати в решті статті, а також представляємо набір цифрових кільцевих осциляторів, які ми оцінили. У розділі 3 ми пропонуємо метод для оцінки випадковості TRNG на технології FPGA; цей метод потім використовується для вибору найкращої архітектури як компроміс між ентропією та складністю. У розділі 4 обране джерело ентропії додатково характеризується відповідно до рекомендацій NIST, і показані результати реалізації на FPGA для обраного генератора випадкових чисел. Завершуючи, у розділі 5 ми підводимо підсумок цієї роботи

$$I_X(x) = -log p_X(x) = log \frac{1}{p_X(x)} \qquad (1)$$

Ця величина є мірою того, наскільки ймовірно, що подія стане результатом випадкового експерименту змінної X. З цього рівняння випливає взаємозв'язок між вмістом інформації події та його ймовірністю: чим рідше відбувається подія, тим більше інформації вона несе з собою. Зазвичай використовується логарифмічна база 2, а одиниця виміру - біт. Найбільш ймовірний результат несе менше інформації, оскільки це найбільш ймовірний результат випадкового експерименту.

Для випадкового джерела інформації (випадкового експерименту) ентропія є мірою середньої кількості бітів інформації, що міститься в сирих даних. Ентропія Шеннона, HS, випадкового експерименту визначається як очікування само-інформації:

$$H_S = E[I_X(x)] = -\sum_i p_X(i) log_2 p_X(i) \qquad (2)$$

Ще одним показником ентропії є Мінімальна Ентропія. Назва містить префікс "Min", оскільки це найстрогіша доступна визначення ентропії. Мінімальна Ентропія випадкового експерименту X визначається як мінімальне значення само-інформації:

$$\begin{aligned} H_{min} &= min[I(x)] = min_i[-log_2(p_i)] \\ &= -log_2[max_i(p_i)] \end{aligned} \qquad (3)$$

Мінімальна Ентропія виражається в бітах і встановлює верхню межу оцінки можливості атакувача передбачити результат. Одна з властивостей Мінімальної Ентропії полягає в тому, що вона ніколи не перевищує Ентропію Шеннона, і вони співпадають, якщо X має рівномірну ймовірнісну функцію маси.

У цьому розділі наводиться огляд основних архітектур, запропонованих в літературі для реалізації цифрових генераторів випадкових чисел (RNGs), зосереджуючись на тих, що підходять для технології FPGA.

Перехідний Кільцевий Осцилятор Ефекту (Transition Effect Ring Oscillator, TERO) [18,22] отримує випадковість, використовуючи осциляторну метастабільність за допомогою захвату. Його структура

зображена на Рисунку 1. Аналізуючи поведінку схеми для різних вхідних значень, можна помітити, що якщо rst = 1, вихідні порти захвату, b1 і b2, залишаються на рівні нуля, і досягається стабільний стан з a1 = a2 = ctrl. Якщо rst = 0, вентилі AND діють як обхідні вентилі, і петля складається, залежно від входів вентилів XOR, з двох буферів і двох інверторів або чотирьох буферів. У обох випадках досягається стабільний стан, оскільки інвертуючі елементи завжди є парними. Коли ctrl переходить з низького на високий або з високого на низький рівень, a1 і a2 інвертуються, що генерує коливання, яке працює всередині зворотного зв'язку, поки захват не стабілізується до постійного стану. Цю поведінку називають осциляторною метастабільністю, про яку широко розповідається в [23]. Осциляторна метастабільність може спостерігатися в захватах з більш ніж двома вентилями і ланцюгом зворотного зв'язку із затримкою. Враховуючи внутрішній шум у напівпровідниках, кількість коливань для кожного переходу на вході ctrl є випадковою величиною.

Ентропія накопичується шляхом підрахунку кількості коливань, і для кожного переходу ctrl генерується один біт: логічна 1 генерується, якщо кількість коливань на вихідних портах, b1 і b2, непарна; логічний 0 генерується, якщо кількість коливань на вихідних портах, b1 і b2, парна. Для цього може використовуватися асинхронний лічильник, розміщений на виході структури TERO: достатньо одного T-тригера для відокремлення непарних та парних коливань, що призводить до досить компактного елементу ентропії. Однак поведінка осциляторної метастабільності не є особливо швидкою: у [18] автор пропонує період ctrl, рівний 4 мкс, що перекладається в пропускну здатність 250 кбіт/с. Особлива увага повинна бути приділена маршрутизації, особливо в балансі між гілками петлі. Як стверджується в [23], асиметрія у часі поширення гілок зворотного зв'язку допомагає зменшити час вирішення: це означає можливість досягнення більшої пропускної здатності, але також ймовірну втрату якості через зменшене середнє квадратичне відхилення випадкової величини, яка

моделює коливання. Ці розрахунки сильно залежать від технології, в якій буде побудований елемент TERO.

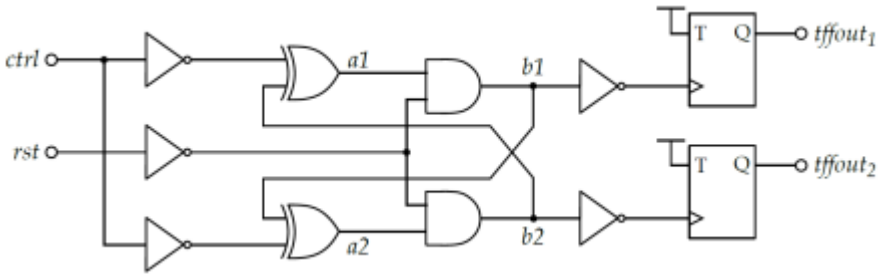


Рисунок 1. Кільцевий осцилятор із ефектом переходу

Метастабільний Кільцевий Осцилятор (Meta-RO) [19] - це ще один варіант кільцевого осцилятора, який збирає ентропію з аналогової метастабільності вентильних гейтів. CMOS інвертор досягає метастабільного стану, якщо його вихід замиканий на вхід із закритим перемикачем. У цій конфігурації вихідна напруга коливається навколо метастабільного рівня через накладений шумовий внесок. Повна схема показана на Рисунку 2. Робота осцилятора розвивається через різні фази:

• Ініціалізація: осцилятор переводиться у метастабільний стан з mux_sel = 0. Низькочастотний шум збурює метастабільну напругу.

• Перехід: сигнал mux_sel переходить у логічне 1, і кільце знову утворюється. Низькочастотний шум посилюється інверторами. Кільце починає працювати у випадковому стані.

• Вибіркове дискретизування: як тільки осцилятор стабілізується до повнологічних рівнів, може відбуватись дискретизація за допомогою D-тригера. Для точного контролю моменту дискретизації автор пропонує генерувати clk з сигналу mux_sel за допомогою лінії затримки.

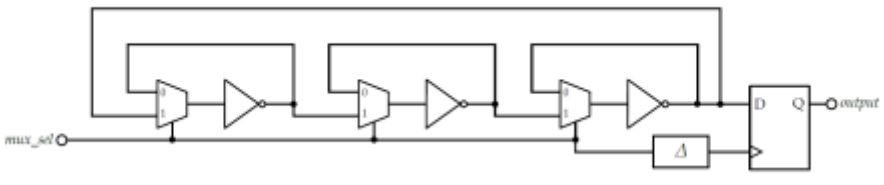


Рисунок 2. Метастабільний кільцевий осцилятор

Перш ніж представляти Кільцеві Осцилятори Фібоначчі та Галуа, варто ввести поняття про лінійні регістри зсуву зі зворотним зв'язком, структури яких надихнули ці Генератори Справжніх Випадкових Бітів (TRBGs). Лінійний Регістр Зсуву (LFSR) - це регістр зсуву з мережею зворотнього зв'язку, зазвичай деревом XOR, яке подає вхід з лінійною функцією поточного внутрішнього стану. Ступінь m LFSR дорівнює довжині самого регістру зсуву.

Два різних архітектурних варіанти LFSR - Фібоначчі та Галуа - зображені на Рисунках 3 та 4. У першому випадку виходи D тригера (DFF), які замикані у зворотній дії перемикачами, XORуються між собою та підключаються до послідовного входу регістру зсуву.

Другий варіант презентує іншу концепцію: якщо fr−i дорівнює 0 (перемикач відкритий), DFFi подає сигнал на DFFi+1, як у звичайній операції регістра зсуву; навпаки, якщо fr−i дорівнює 1 (перемикач закритий), вихід DFFi XORується з послідовним виходом регістру зсуву для створення входу для DFFi+1.

У обох випадках функцію зворотнього зв'язку контролюють перемикачі fi, які зазвичай залишаються закритими, якщо fi = 1, або відкритими, якщо fi = 0. Ці значення можуть бути розглянуті як коефіцієнти полінома: характеристичного полінома LFSR.

$$P(x) = 1 + f_1 x + f_2 x^2 + \dotplus f_m x^m \qquad (4)$$

Оскільки новий стан залежить лише від попереднього, а кількість станів обмежена (фактично, вони обмежені довжиною Регістру Зсуву), кожен LFSR видає періодичний шаблон.

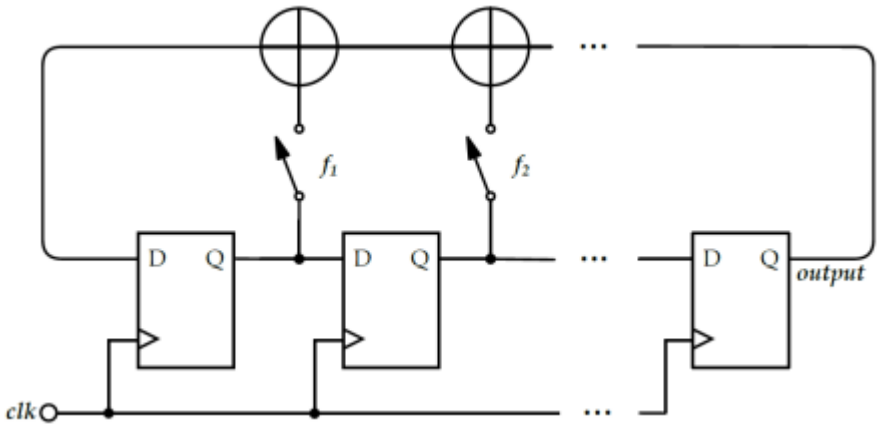


Рисунок 3. Лінійні регістри зсуву Фібоначчі.

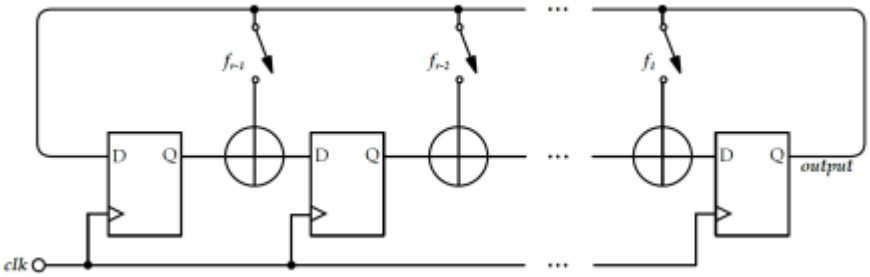


Рисунок 4. Лінійний регістр зсуву Галуа

За допомогою LFSR можна створювати псевдовипадкові послідовності з набагато більшим періодом, ніж довжина вихідної послідовності. Однак жодної ентропії ніколи не генерується всередині весьма визначеного кола, подібного цьому, і легко довести, що LFSR не виявляють ні передбачуваності, ні стійкості до відстеження. Зміни в цих двох архітектурах були внесені в [24], де були представлені дві нові архітектури TRNG: Кільцеві Осцилятори Фібоначчі (FiROs) та Кільцеві Осцилятори Галуа (GaROs).

FiRO походить від своєї аналогічної архітектури LFSR, замінюючи всі тригери D інверторами, як показано на Рисунку 5. Оскільки всі синхронні елементи вилучені, коло є повністю асинхронним та розвивається в стохастичний спосіб. Випадковість вводиться завдяки залежності затримки інвертора від температури та напруги живлення: при зміні цих параметрів, як для шуму, так і для змін середовища, кільце розвивається з різними вихідними шаблонами, що призводить до хаотичного та непередбачуваного

сигналу. Додаткова випадковість може бути отримана під час фази вибірки: порушення часів налаштування та утримання може призвести до метастабільності в блоках вибірки. FiRO складається з r елементів інверторів, що послідовно з'єднані, так що кожен з них (крім останнього) створює вхід для наступного інвертора. Функція зворотнього зв'язку визначена, як і в LFSR, коефіцієнтами fi, які можуть бути представлені за допомогою характеристичного полінома:

$$P(x) = \sum_{i=0}^{r} f_i x^i \qquad with \quad f_0 = f_r = 1 \tag{5}$$

Для забезпечення достатньої випадковості важливо уникати фіксованих точок, щоб коливання ніколи не припинялося. Фіксованих точок не має, якщо виконуються ці дві умови:

$$\begin{cases} P(x) = (x+1)h(x) \\ h(1) = 1 \end{cases} \tag{6}$$

GaRO походить від структури Galois LFSR, знову замінюючи всі тригери D інверторами, як показано на Рисунку 6. Ті ж самі розгляди, які були проведені з FiRO щодо стохастичної еволюції цього кола, залишаються актуальними. GaRO складається з r інверторів, які послідовно з'єднані так, що вихід кожного інвертора є входом для вентиля XOR, який утворює вхід наступного інвертора.

Функцію зворотнього зв'язку можна представити за допомогою характеристичного полінома:

$$P(x) = \sum_{i=0}^{r} f_i x^i \qquad with \quad f_0 = f_r = 1 \tag{7}$$

Умови, за яких GaRO не має фіксованих точок, наступні:

$$\begin{cases} P(1) = 0 \\ r \quad is \quad odd \end{cases} \tag{8}$$

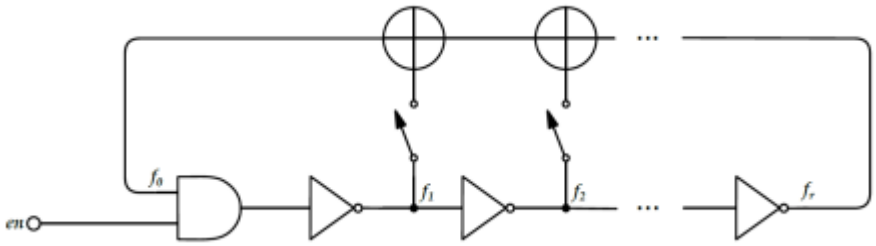
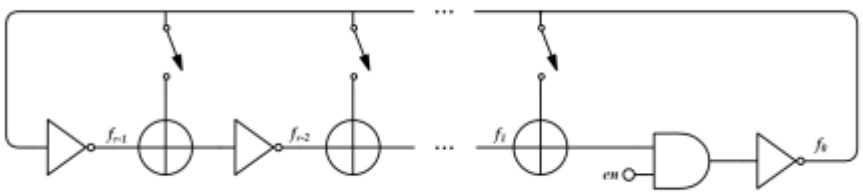


Рисунок 5. Кільцевий осцилятор Фібоначчі з портом увімкнення.



Фігура 6. Кільцевий осцилятор Галуа з портом увімкнення.

Випадковість, а також стійкість, можна ще більше збільшити, XOR-уючи виходи Кільцевого Осцилятора Фібоначчі та Кільцевого Осцилятора Галуа. Структура, утворена підключенням FiRO та GaRO за допомогою операції XOR, отримала назву FiGaRO. На Рисунку 7 показано приклад структури FiGaRO, яка була зазначена в [24], що складається з чотирьох FiRO та чотирьох GaRO, підключених за допомогою дерева XOR. Довжина обох осциляторів, за винятком одиниці, повинна бути бажано взаємно простою, щоб максимізувати період відповідної псевдовипадкової послідовності та зменшити зв'язок. Крім того, рекомендується, щоб довжини відрізнялися лише на одиницю, де парна довжина відповідає Кільцевому Осцилятору Фібоначчі.

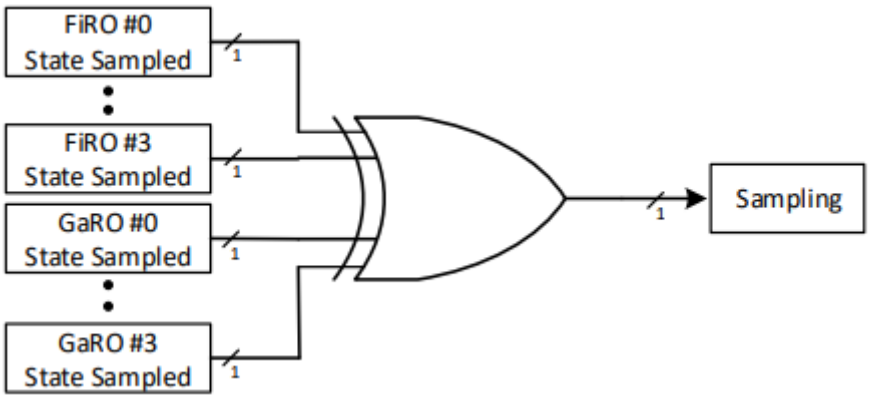


Рисунок 7. Архітектура FiGaRO (Кільцевий Осцилятор Фібоначчі-Галуа) з чотирма елементами

3.3. Проектування та Оцінка Генераторів Справжніх Випадкових Чисел на Технології FPGA

Методи аналізу Генераторів Справжніх Випадкових Чисел на основі технології FPGA

Як було вказано у попередніх розділах, ГСВЧ використовує різні джерела шуму для генерації випадкового виводу. Ця особливість підказує, що функціональне тестування цих пристроїв не є простим завданням. Функціональні симуляції на рівні трансферу регістрів (RTL) не підходять, оскільки вони не можуть розв'язати комбінаційну петлю, яка присутня практично в кожному сучасному ГСВЧ. Після-розміщувальні симуляції можуть це зробити, але події метастабільності, які є фундаментальною частиною процесу збору ентропії, призводять до поширення в ланцюзі вибірки не визначеного логічного стану (X). Аналогові інструменти симуляції надають функціональні можливості для аналізу шуму в області часу, і можна знайти в літературі статті, де вони використовуються для симуляції ГСВЧ на спеціалізованих інтегральних мікросхемах (ASIC) [19]. Однак наша основна ціль для реалізації - це FPGA, і вендори не надають моделей для тестування динамічної поведінки комбінованої петлі, залежної від шуму. З цих причин тестування проводилося безпосередньо на обладнанні. Наша стратегія тестування спрямована на пряме отримання набору випадкових значень на FPGA, які потім передаються на зовнішній комп'ютер для обробки та оцінки.

У тестовій кампанії, про яку йдеться у цьому розділі, ми провели тестування вже запропонованих архітектур ГСВЧ (TERO, Meta-RO, FIRO, GARO та FiGaRO). Метою цієї тестової кампанії є пошук найкращої архітектури для FPGA, де основним показником є випадковість виводів. Працюючи на FPGAs, ми спробували випробувати різні методи розміщення апаратного

забезпечення, щоб знайти архітектуру, випадковість якої не залежить від розташування апаратного забезпечення в межах FPGA. Оцінка рівномірності розподілу здійснюється кількісно на першому етапі. Вибраний ГСВЧ потім буде протестований з використанням тестового комплекту оцінки ентропії NIST для точної оцінки його якості. Вплив різних варіантів розміщення та маршрутизації на продуктивність деяких пристроїв може бути великим. Де потрібно, в інструментах розробки FPGA були створені обмеження для вирішення цих проблем: використання функцій Розбиття Дизайну та LogicLock, розміщення та маршрутизацію одного елемента ГСВЧ, відображеного в обмеженому числі логічних блоків масиву (LABs), було збережено для повторного виклику декілька разів та розміщено в різних комірках, які контролюються користувачем. Такі обмеження можна легко використовувати в будь-якому інструменті розробки FPGA. Таким чином, було можливо випробувати різні конфігурації планування та отримати більш точні результати.

Способи вирішення

Кільцевий Осцилятор Ефекту Переходу (TERO)

Для тестування TERO, представленого у Розділі 2.2.2, необхідно оцінити кількість осциляцій, які виконує зачіпка кожного разу, коли сигнал ctrl перемикається. Для цього виведено 10-бітний асинхронний лічильник на виході пристрою. Використовуючи 10-бітний лічильник замість одного Т-тригера, ми визначаємо, чи кількість осциляцій парна чи непарна, і можемо перевірити точну кількість осциляцій. Очікуваним результатом є отримання розподілу кількості осциляцій, схожих на ті, що показані в [25,26]. Фактична випадкова величина виводиться у двійковій формі: одиниця, якщо кількість осциляцій є непарним числом, або нуль, якщо вона є парним числом. Вибірки можна розглядати як незалежні, оскільки TERO виводить один біт для кожного перезапуску, що усуває можливі залежності між послідовними вибірками. Екземпляри, де кількість осциляцій різниться більше, надають менше різниці між одиницями і нулями, тобто вищу ентропію. Ми

розмістили кілька TERO на FPGA з однаковим внутрішнім розміщенням і маршрутизацією, обмежені в одній групі з 10 модулів адаптивної логіки (ALMs). Цей елемент був розміщений кілька разів, щоб перевірити, чи залежить спостережене поведінка від розміщення всередині пристрою FPGA. З кожного TERO було отримано послідовність з 10 000 чисел осциляцій, і були побудовані гістограми цих значень. На Рисунку 8 показано кількість осциляцій, які відбулися до стабілізації виходу зачіпки, в 8 різних елементах TERO. Як показано, гістограми кількості осциляцій дуже відрізняються від одного елемента до іншого: деякі, наприклад TERO 5, майже мають фіксовану кількість осциляцій, тоді як інші, наприклад TERO 4, показують більш складний зразок. Ця змінність була виявлена також в [26]. Автори у [26,27] стверджують, що залежність від маршрутизації та розміщення є основним недоліком TERO. Техніки для подолання цієї проблеми були

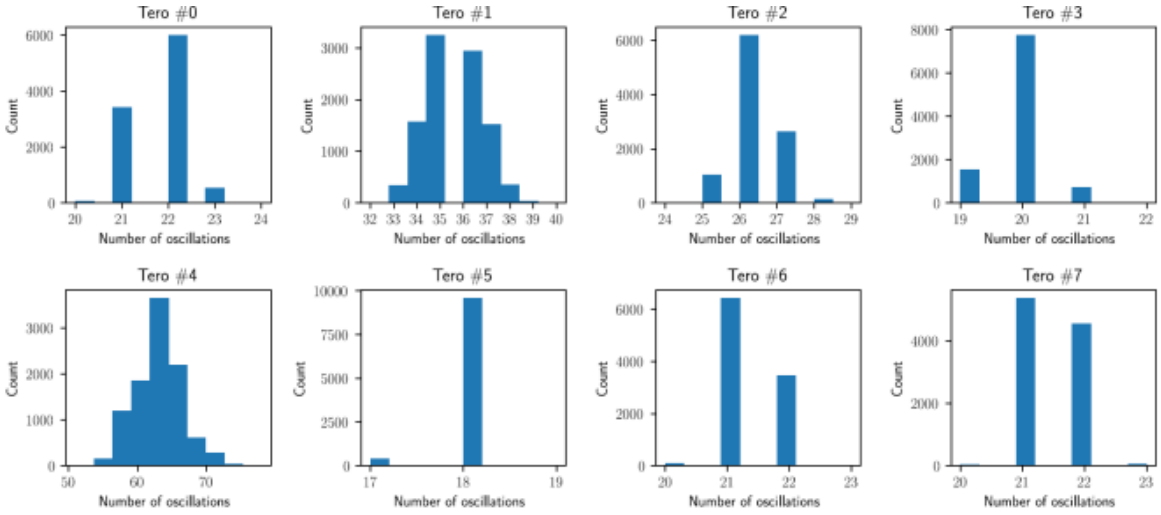


Рисунок 8. Гістограма кількості осциляцій для 105 подій спрацювання для Генератора випадкових чисел на основі кільцевого осцилятора ефекту переходу (TERO).

Метастабільний кільцевий осцилятор (Meta-RO)

Meta-RO потребує надійної лінії затримки між сигналом активації та тактовим сигналом збору вибірки. Цю затримку можна отримати, використовуючи PLL, який був сконфігурований для вироблення сигналу активації з такою ж частотою як тактовий сигнал (100 МГц) і фазовим зсувом

-1 нс. У нашому випадку, PLL, який використовується, є апаратним примітивом FPGA.

На відміну від TERO, у елементів Meta-RO кількість осциляцій не є значущою для статистики виводу. Тому тестування відбувалося трохи по-іншому: восьмох однакових заблокованих елементів Meta-RO розміщувалися в різних лабораторіях логічного блоку (LAB), а паралельне вихідне значення, інтерпретоване як 8-бітне ціле число (Значення вибірки, 0–255), оцінювалося, як показано на Рисунку 9. Цей тест повторювався кілька разів, генеруючи нові послідовності бітів для розуміння залежностей від розміщення та маршрутизації. Кожен запуск отримав 131,072 вибірки з восьми паралельних елементів Meta-RO. Рисунок 10 показує гістограми вихідних значень: навіть якщо показники самого осцилятора можна вважати задовільними (як показано в [28]), дані далекі від рівномірного розподілу, а форма пилоуса підкреслює перевагу нулів над одиницями, що відображається в виводі низької якості.

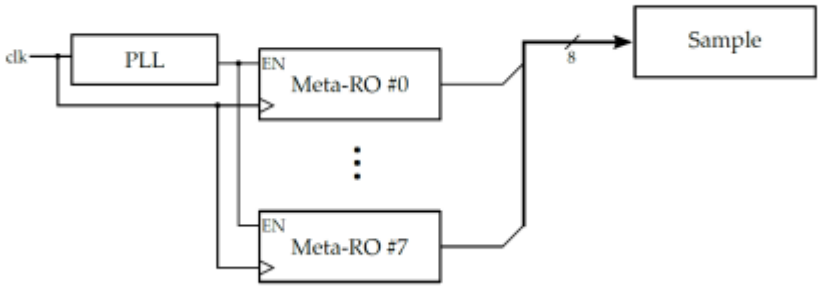


Рисунок 9. Стратегія вибірки для оцінки Meta-RO.

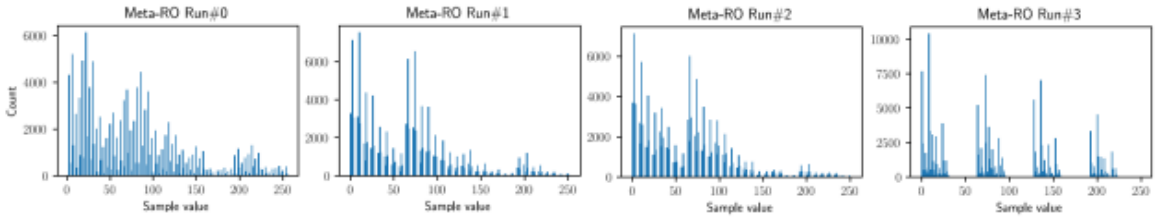


Рисунок 10. Гістограми вихідних значень для восьми паралельних елементів Meta-RO.

FiRO and GaRO

Кілька варіантів зворотних поліномів можуть бути вибрані для FiRO і GaRO. Один з можливих варіантів включає той, що був представлений у [20], де всі мережі зворотного зв'язку FiRO і GaRO були окремо протестовані на технології Intel FPGA Cyclone V для довжин осциляторів, відповідно, рівних 10 та 11. Поліноми, визначені як найкращі у [20], були обрані для реалізації джерела ентропії, та була також додана підтримка збору стану для взяття вибірки не лише остаточного інвертора, але й усіх внутрішніх станів TRNG. На Рисунках 11 та 12 показана структура FiRO GaRO, обидва зі збором стану.

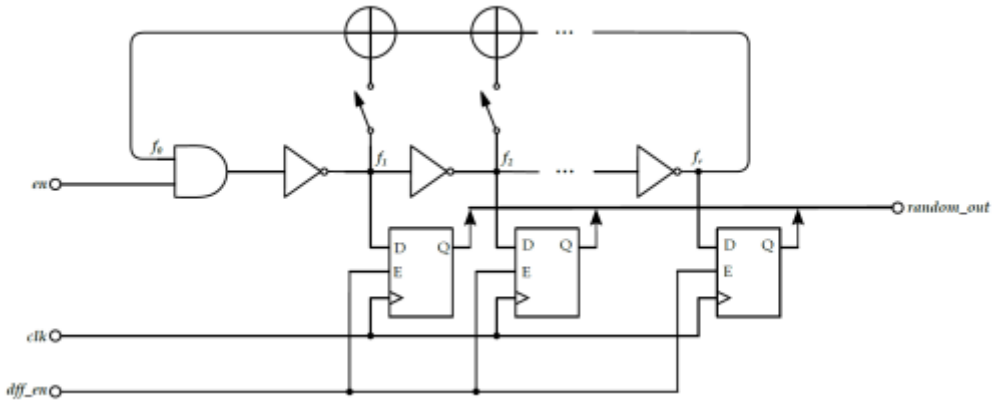


Рисунок 11. Кільцевий осцилятор Фібоначчі зі збором стану. Вихідна шина XOR-ується разом, щоб утворити 1-бітовий вихід на вищому рівні.

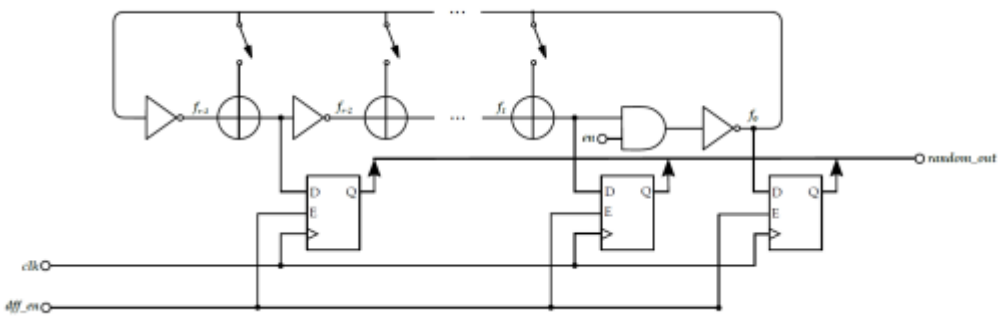


Рисунок 12. Кільцевий осцилятор Галуа зі збором стану. Вихідна шина XOR-ується разом, щоб утворити 1-бітовий вихід на вищому рівні.

Перша ітерація тестування полягала у створенні чотирьох FiRO і чотирьох GaRO з наступними конфігураціями зворотного зв'язку, і зчитування їх восьмибітового паралельного виводу 131,072 рази. Зверніть увагу, що F позначає FiRO, а G - GaRO.

$$p_0^F(x) = 1 + x + x^2 + x^3 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10} \quad (9)$$

$$p_1^F(x) = 1 + x + x^2 + x^3 + x^4 + x^6 + x^7 + x^{10} \quad (10)$$

$$p_2^F(x) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^{10} \quad (11)$$

$$p_3^F(x) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^9 + x^{10} \quad (12)$$

$$p_0^G(x) = 1 + x^2 + x^3 + x^5 + x^6 + x^7 + x^9 + x^{11} \quad (13)$$

$$p_1^G(x) = 1 + x^2 + x^3 + x^5 + x^6 + x^7 + x^8 + x^{11} \quad (14)$$

$$p_2^G(x) = 1 + x^2 + x^3 + x^4 + x^5 + x^7 + x^9 + x^{11} \quad (15)$$

$$p_3^G(x) = 1 + x^2 + x^3 + x^4 + x^5 + x^6 + x^9 + x^{11} \quad (16)$$

Не надано жодних особливих обмежень для частоти відбору та розміщення та маршрутизації. Порівнюючи запуск збору даних, зображений у Таблиці 2, з тими, що отримані для Meta-RO, стає очевидним зменшення спрямованості, яке підтверджується кількістю одиниць та нулів.

	FiRO Instance				GaRO Instance			
	#0	**#1**	**#2**	**#3**	**#0**	**#1**	**#2**	**#3**
Count of zeros	64,463	62,237	62,336	62,624	6274	65,784	64,161	58,209
Count of ones	66,609	68,835	68,736	68,448	68,308	65,288	66,911	72,863
Difference	−2146	−6598	−6400	−5824	−5544	496	−2750	−14,654

Таблиця 2. Кількість одиниць та нулів у послідовності виведення FiRO-GaRO.

FiGaRO

Структура, зображена на Рисунку 7, використовується для формування одного вихідного біта, і вона була реплікована вісім разів для отримання 8-бітового паралельного виводу. Від цієї структури очікується вища ентропія, що підтверджується експериментальними результатами: одна й та ж кількість отриманих зразків призвела до більш рівномірного розподілу, як показано на Рисунку 13. Як показано в Таблиці 3, спостерігається дуже хороший баланс між кількістю одиниць та нулів, а спрямованість, на перший погляд, здається випадковою і фізіологічною через обмежений набір даних, який ми розглядаємо.

	FiGaRO Instance							
	#0	**#1**	**#2**	**#3**	**#4**	**#5**	**#6**	**#7**
Count of zeros	65,543	65,716	65,256	65,512	65,566	65,777	65,335	65,678
Count of ones	65,529	65,356	65,816	65,560	65,506	65,295	65,737	65,394
Difference	14	360	−560	−48	60	482	−402	284

Таблиця 3. Кількість одиниць та нулів у послідовності виведення FiGaRO

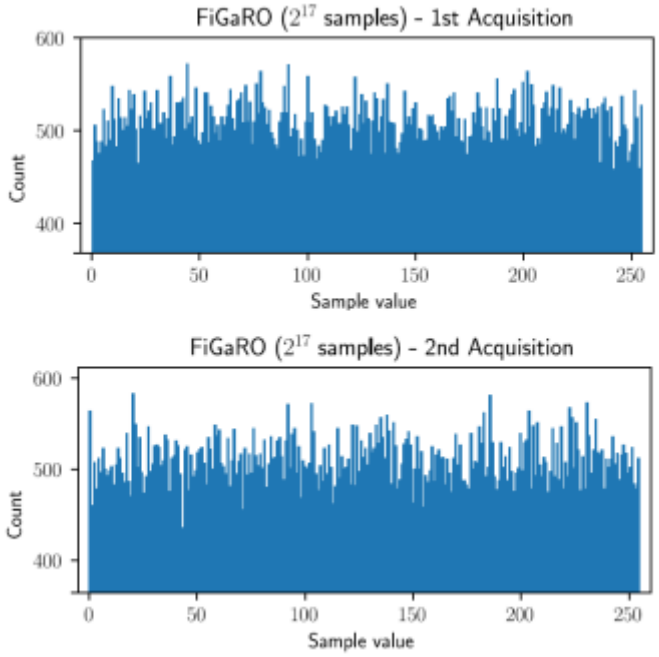


Рисунок 13. Гістограми першого та другого запусків збирання даних для FiGaRO TRNG.

3.4 Проектування FiGaRO RNG

Архітектура FiGaRO, яка вже довела свою перевагу над класичними кільцевими осциляторами [21], продемонструвала свою перевагу серед усіх інших представлених осциляторів щодо ентропії для пристрою FPGA. Як тільки FiGaRO було обрано як джерело ентропії, ми перейшли до проектування повного генератора випадкових чисел, який включає також секції управління та контрольні схеми для перевірки працездатності. Після цього система була протестована на відповідність до тестового набору NIST, щоб підтвердити відповідність відповідним стандартам. У цьому розділі ми покажемо, як були спроектовані блоки випадкового числового генератора (RNG) у SystemVerilog та реалізовані в технології FPGA. Проект RNG

повинен охоплювати як пристрій збору ентропії (тобто TRNG), так і контрольні схеми для перевірки працездатності.

Тести на стійкість

Звідусіль, в усіх джерелах ентропії потрібно постійно перевіряти статистичну якість виводу, щоб виявити значні зміни, спричинені різними факторами, такими як зміна умов навколишнього середовища, електричні впливи або можливі атаки. Щоб уникнути ситуацій, коли ентропія занадто низька, важливо швидко виявляти будь-які несправності, тимчасово зупиняючи вивід, поки якість не відновиться. Згідно рекомендаціям NIST у [29], для джерела ентропії потрібні два онлайн-тести на стійкість: тест на рахунок повторів і тест на адаптивний відсоток, які описані нижче.

Тест на рахунок повторів спрацьовує тривалою послідовністю однакових значень виводу. Для проведення цього тесту потрібно вибрати рівень помилкової ймовірності α: чим вище цей рівень, тим менше повторень потрібно для спрацювання тривоги. Одразу після вибору α кількість повторень, що викличуть тривогу, визначається за наступною формулою:

$$n = 1 + \left\lceil \frac{-log_2(\alpha)}{H} \right\rceil \qquad (17)$$

Пропонується апаратна реалізація цього тесту для двійкових джерел: зсувний регістр безперервно збирає виходи джерела ентропії, а додаткова логіка перевіряє, чи паралельний вихід зсувного регістру складається виключно з одиниць або лише з нулів. Ця реалізація показана на Рисунку 14.

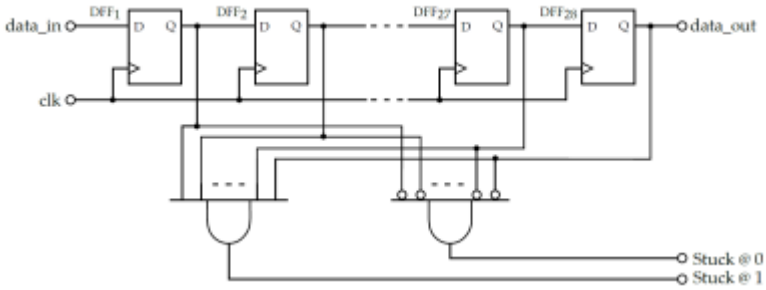


Рисунок 14. Апаратна реалізація тесту на повторення для бінарного джерела випадкових чисел.

На базі адаптивного тесту пропорції реалізується перевірка частоти випадання кожного можливого значення виходу в межах певного довірчого інтервалу. Якщо вивід з'являється занадто часто, це свідчить про зниження статистичної якості джерела випадковості, і слід викликати сповіщення. Для двійкових виводів тест повинен бути проведений на вікні з 1024 вибірками. Кількість одиниць має бути порахована і порівняна з двома різними значеннями: якщо кількість одиниць перевищує величину C або менше за (1024 − C), слід підняти сповіщення. Величини C обчислюються за допомогою NIST як функція ентропії послідовності бітів. Знову ж таки, запропоновано апаратну реалізацію цього тесту, зображену на рисунку 15. Регістр зсуву довжиною 1024 біти зберігає пам'ять останніх виводів, і кожного разу, коли приходить нове значення, його значення додається до реєстра акумулятора, а останнє значення віднімається. Таким чином, акумулятор завжди містить кількість одиниць в межах вікна спостереження, і два порівняльники можуть перевірити, чи частота лежить в межах верхньої та нижньої меж.

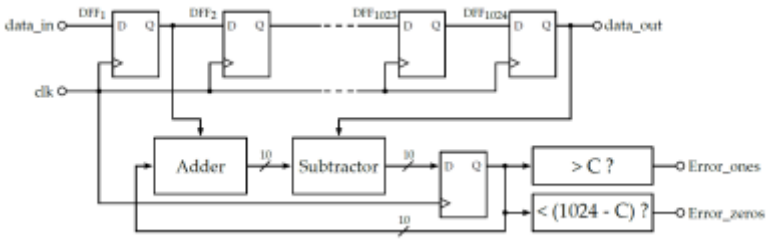


Рисунок 15. Апаратна реалізація тесту адаптивної пропорції для джерела випадкових двійкових значень.

3.5 Загальна архітектура Генератора Випадкових Чисел (ГВЧ)

На рисунку 16 показана архітектура запропонованого ГВЧ. FiRO та GaRO, налаштовані як FiGaRO, були вибрані як джерело ентропії. Наш внесок до цього полягає у включенні тесту на стан здоров'я та комбінації

параметризованої кількості етапів, для збільшення пропускної здатності. Така архітектура показала себе найбільш придатною для реалізації на FPGA завдяки незалежності від розташування та маршрутизації у FPGA. Остаточний дизайн був параметризований для можливості налаштування кількох етапів (тобто кількість паралельних ГВЧ): один, два, чотири та вісім елементів із виводом 1 біт можуть бути розміщені паралельно. Кожен етап називається етапом FiGaRO, як показано на рисунку 16. Зверніть увагу, що кількість FiRO та GaRO, розміщених у FiGaRO, сприяє покращенню ентропії та надійності: насправді, якщо один або кілька осциляторів застрягне, все ще може працювати ГВЧ, оскільки окремі осцилятори об'єднуються за допомогою операції XOR. З іншого боку, кількість FiGaRO етапів, розміщених паралельно, впливає тільки на збільшення пропускної здатності, платячи лінійно за використані ресурси; цей параметр не впливає на ентропію. Кожен етап має однакову структуру: кілька FiRO та GaRO можуть бути об'єднані за допомогою операції XOR, щоб сформувати високоентропійний вивід, зі зворотними поліномами, зазначеними в Рівняннях (9)–(16). Обираючи різні мережі зворотнього зв'язку, мінімізується кореляція між FiRO та GaRO в межах етапу FiGaRO. Ця операція також забезпечує вищий рівень надійності джерела ентропії, оскільки різні внески сумуються, маскуючи можливі тимчасові відмови деяких осциляторів. Крім того, кількість FiRO та GaRO, які об'єднуються в кожному етапі, параметризована для вибору між одним FiRO та одним GaRO (з поліномами Рівняння (9) та (13)), двома FiRO та двома GaRO (з поліномами Рівняння (9), (10), (13).

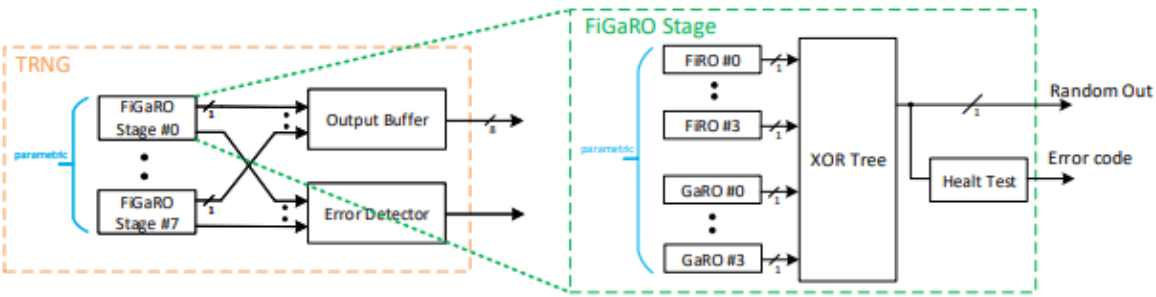


Рисунок 16. Загальна архітектура реалізованого генератора випадкових чисел FiGaRO (RNG).

Попередній аналіз, представлений у розділі 3.2, був проведений якісно, зосереджуючись передусім на оцінці емпіричних результатів, отриманих з різних архітектур TRNG. Тим не менш, була необхідність перевірити обрану архітектуру за допомогою схвалених тестів, які могли б забезпечити доступну оцінку ентропії цього пристрою.

Вимірювання ентропії FiGaRO RNG

Для оцінки якості послідовності, що генерується джерелом ентропії, було необхідно визначити мінімальну ентропію на біт згенерованої послідовності за допомогою специфічних тестів. Ці тести запропоновано і описано в [29], тоді як набір тестів, що базуються на цих характеристиках, був розроблений самим NIST та вільно доступний у [30]. Під час цих тестів було досліджено дві різні речі:

• Перше стосувалося того, як якість виходу залежить від кількості елементів, які були складені операцією XOR всередині кожного етапу FiGaRO. Ідеальною ситуацією було використання більшої кількості елементів, але це залежало від припущення незалежності пристроїв, яке треба було перевірити.

• Друге стосувалося того, як якість виходу залежить від частоти вибірки. До цього моменту відбувалося відбіркове взяття на частоті 1 МГц, але побудова графіка ентропії як функції частоти могла б дати змогу зробити компроміс між пропускною здатністю джерела ентропії та якістю виходу.

Перед проведенням тестування необхідно було зібрати дані у певний спосіб. Зокрема, для кожного тесту надавалась послідовність принаймні з 1 мільйона послідовних вибірок, яку назвали S:

$$S = [s_0\ s_1\ \dots\ s_{999999}] \tag{18}$$

і матрицю перезапуску 1000 × 1000, яку відтепер будемо називати R, побудовану, перезапускаючи джерело ентропії 1000 разів і заповнюючи кожний рядок 1000 послідовними вибірками:

$$R = \begin{bmatrix} S_0^0 & S_1^0 & S_2^0 & \dots & S_{999}^0 \\ S_0^1 & S_1^1 & S_2^1 & \dots & S_{999}^1 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ S_0^{999} & S_1^{999} & S_2^{999} & \dots & S_{999}^{999} \end{bmatrix} \quad (19)$$

З R було створено дві послідовності: набір рядків, який складається з конкатенації всіх рядків R:

$$R_{row} = [s_0^0 \dots S_{999}^0 \, s_0^1 \dots S_{999}^1 \dots S_0^{999} \dots S_{999}^{999}] \quad (20)$$

Набір стовпців, складений із конкатенації всіх рядків R(T):

$$R_{col} = [s_0^0 \dots S_{999}^0 \, s_0^0 \dots S_1^{999} \dots S_{999}^0 \dots S_{999}^{999}] \quad (21)$$

Одного разу побудований набір даних можна було передати до тестового набору статистичних тестів відповідно до [29], щоб перевірити припущення про незалежність та ідентичність розподілу (IID): якщо це припущення підтверджувалося, оцінка ентропії значно спрощувалася, оскільки потрібно було запустити лише один тест (найбільш поширений оцінювання ентропії за найбільш поширеним значенням).

Для проведення частотного скану обрано сім частот вибірки

$$f = [1, 2.5, 5, 10, 25, 50, 100] MHz \quad (22)$$

Далі було протестовано всі три конфігурації FiGaRO Stage, які забезпечували послідовність під тестування:

• 1 FiRO та 1 GaRO, які були XOR-з'єднані
• 2 FiRO та 2 GaRO, які були XOR-з'єднані
• 4 FiRO та 4 GaRO, які були XOR-з'єднані

Це призвело до отримання 21 різних випадків для кожного тестового запуску. Для кожного випадку було необхідно отримати 1 048 576 послідовних вибірок і матрицю перезапуску розміром 1000 × 1000. Було зроблено

твердження про IID для всіх послідовних послідовностей, і тести NIST (тести перестановки та хі-квадрат) підтвердили це твердження. Ентропія, в кінці тестування, становила:

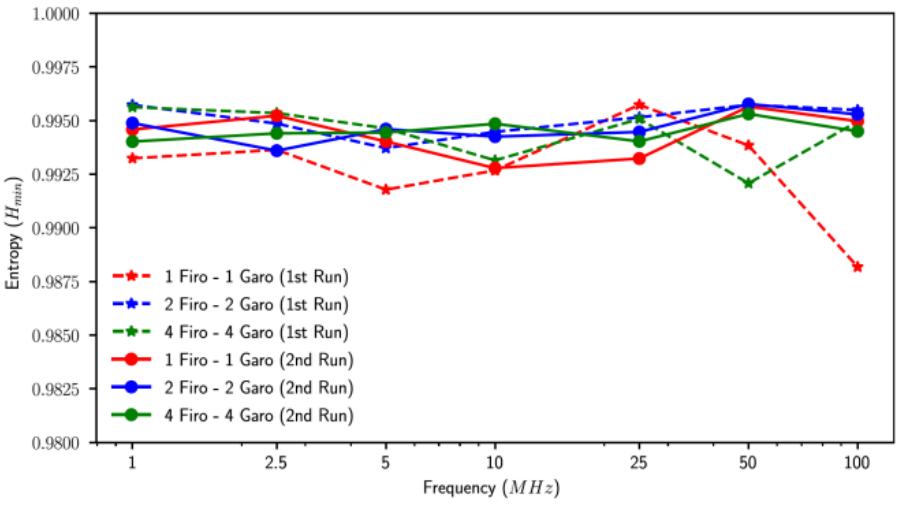
$$H_{min} = min(H_I, H_R, H_C) \tag{23}$$

де H1 - це початкова оцінка ентропії послідовної послідовності, а HR та HC присвоюються даним перезапуску. На рисунку 17 показані значення Hmin, отримані в двох запусках тестів. Всі конфігурації показали високу якість практично незалежно від частоти вибірки, причому всі ентропії стабілізувалися навколо 0,995. Оскільки іноді такі типи результатів представляються за метрикою ентропії Шеннона, нижче наведено перетворення з Hmin на Hc.

$$H_{min} \approx 0.995 \rightarrow \max_i(p_i) = 2^{-H_{min}} = 0.50173587425 \tag{24}$$

$$H_s = -2^{-H_{min}} \log_2(2^{-H_{min}}) - (1 - 2^{-H_{min}}) * \log_2(1 - 2^{-H_{min}}) = 0.9999913 \tag{25}$$

Зверніть увагу, що лише конфігурація 1FiRO-1GaRO мала трохи нижчий рівень ентропії. Конфігурації 2FiRO-2GaRO та 4FiRO-4GaRO були схожі за рівнем ентропії.



"Рисунок 17. Результати оцінки ентропії (два різні запуски збирання даних)"

Синтез та результати реалізації

У Таблиці 4 наведені результати реалізації, отримані на FPGA Stratix IV. Робоча частота синтезу була фіксована на рівні 100 МГц, тому кожен етап FiGaRO вніс свій внесок у пропускну здатність на рівні 100 Мбіт/с; таким чином, ми досягли пропускної здатності на рівні 400 Мбіт/с для чотирьохступеневого осцилятора та 800 Мбіт/с для восьмиступеневого осцилятора. Отримані цифри показали, як очікувалося, що використання ресурсів зростало лінійно як з цими, так і з іншими параметрами.

FiGaRO Stages	Configuration	Comb. ALUTs	Logic Regs
8	1FiRO 1GaRO	291	186
8	2FiRO 2GaRO	571	354
8	4FiRO 4GaRO	1140	698
4	1FiRO 1GaRO	148	106
4	2FiRO 2GaRO	288	190
4	4FiRO 4GaRO	572	358
2	1FiRO 1GaRO	76	62
2	2FiRO 2GaRO	146	104
2	4FiRO 4GaRO	288	188
1	1FiRO 1GaRO	40	40
1	2FiRO 2GaRO	75	61
1	4FiRO 4GaRO	146	103

Таблиця 4. Результати реалізації ГСЧ на програмованому Stratix IV

• Конфігурація 1FiRO-1GaRO мала найменший відбиток площі, але також мала найгіршу ентропію.

• 2FiRO-2GaRO та 4FiRO-4GaRO були порівняні за ентропією.

• Системна стійкість до можливого застрягання в осциляції зростала лінійно з кількістю FiRO та GaRO, які складають стадію FiGaRO.

• Використання ресурсів зростало лінійно з кількістю FiRO та GaRO, які складають стадію FiGaRO.

• Пропускна здатність системи зростала лінійно з кількістю паралельних стадій FiGaRO.

• Використання ресурсів зростало лінійно з кількістю паралельних стадій FiGaRO.

Таблиця 5 показує порівняння між нашою роботою та іншими ГВПГ, реалізованими на платформах FPGA. Зверніть увагу, що ми не втілили звітований дизайн, але спиралися на результати, представлені у цитованих роботах. З тієї ж причини і для лаконічності ми тут не наводимо архітектурні деталі дизайну; їх можна знайти, переглянувши зазначені роботи. На жаль, доступні роботи базувалися на різних вузлах технології FPGA: це зробило непорівнянним порівняння у термінах рівня бітів, оскільки новіші технології, очевидно, можуть досягати вищих частот тактового сигналу. Однак інші метрики не були безпосередньо уражені різницею в технологічних вузлах. Крім того, більшість запропонованих рішень (наприклад, ті, що базуються на саморегулювальних кільцях - STRs) не були обмежені технологією, але скоріше методом збору ентропії: частота тактового сигналу була обмежена значно нижчими значеннями, ніж максимально доступні, для збереження якості випадковості. Огляд порівняння все ж надав вказівку щодо досяжної пропускної здатності випадкових чисел, разом з якістю випадковості (ентропією).

	TRNG Type	Platform	LUTs	Registers	Bit Rate (Mbps)	Entropy
This work (4 Stages, 2FiRO-2GaRO)	FiGaRO	Intel Stratix IV	288	190	400	0.995
[31]	ES	Xilinx Spartan 6	10	5	1.15	0.997 (Shannon Entropy)
[32]	RO	Xilinx Virtex 2	–	–	2.5	0.97
[33]	RO—PDLs	Xilinx Spartan-3A	528	177	6	0.9993
[34]	STRs	Xilinx Virtex 6	32	48	4	–
[27]	TERO	Xilinx Artix 7	40	29	1.91	0.9993 (Shannon Entropy)
[35]	STRs	Xilinx Virtex 6	56	19	100	–
[36]	GaRO	Xilinx Artix 7	50	79	280	0.998 (Shannon Entropy)

Таблиця 5. Порівняння запропонованого ГВПГ з іншими реалізаціями ГВПГ на платформі FPGA.

В [27] автори демонструють, що TERO сильно залежить від місця розташування; тому вони пропонують особливий TERO (TC-TERO), який подолав цю проблему, використовуючи залежні від FPGA примітиви, але втрачаючи тим самим технологічну незалежність. Автори в [34] запропонували реалізацію, засновану на явищі координованого зразка (CS), використовуючи самозбуджувані кільцеві генератори (STRs) замість зазвичай використовуваних кільцевих осциляторів (ROs). Автори продемонстрували переносимість дизайну та досягли дуже компактного рішення, але швидкість передачі була обмежена низькою частотою зразка. Насправді, дизайн був синтезований з частотою 1 МГц на FPGA Xilinx Virtex 6, і автори стверджують, що вищі частоти зразка можуть погіршити якість випадковості. Натомість, як показано на Рисунку 17, у нашому запропонованому рішенні ентропія стабільна змінюється з частотними коливаннями. Дизайн, запропонований у [31], заснований на техніці зразкового вибору крайніх значень (ES) та досягає найнижчого споживання ресурсів. Пропускна здатність залишається обмеженою на рівні 1,15 Мбіт/с та 1,07 Мбіт/с відповідно на FPGA Xilinx Spartan-6 і Intel Cyclone V FPGA, з Шеннонівською ентропією 0,997; згідно з авторами, ця ентропія не може бути досягнута для вищих частот без спеціалізованих ланцюжків обробки після збору даних. У роботах [32,33] використовувалися RO як джерела випадковості. Зокрема, автори в [33] включили програмовані лінії затримки (PDLs) для генерації великої кількості осциляцій та введення відхилень, що дозволяє досягнути високої ентропії. Навіть якщо досягнута ентропія вражає, використання ресурсів не є зовсім знехтуваною, і, крім того, результат сильно залежить від робочої частоти, що є обмеженням, відсутнім у нашому запропонованому ГВПГ. У [35] автори пропонують ще одну реалізацію на основі STRs. Співвідношення ресурсів до швидкості передачі даних виявилося значно вищим, ніж у [34]; проте їх важко подальше порівняти через відсутність виміряної ентропії. Нарешті, в [36] автори пропонують дуже конкурентоспроможне рішення за ентропією, складністю та

використанням ресурсів. Недолік тут полягає в тому, що використання лише GaRO робить осцилятор більш схильним до несправностей порівняно з підходом FiGaRO [24]. Наша робота показала дуже високу пропускну здатність з використанням ресурсів на рівні із іншими роботами та високий рівень ентропії на біт. Крім того, наша реалізація легко адаптується до потреб користувача, сплачуючи використанням ресурсів лінійно в залежності від потрібного рівня надійності та необхідної пропускної здатності. Щодо ентропії, ми рекомендуємо використовувати конфігурацію 1FiRO-1GaRO лише для оптимізації ресурсів, інакше, оскільки необхідне обладнання все ще обмежене, можна досягти вищої ентропії з іншими двома конфігураціями.

Якість спроектованого джерела ентропії було оцінено за допомогою широко використовуваного набору статистичних тестів NIST 800.22 [37]. Він складається з 15 типів тестів, які були виконані для оцінки продуктивності TRNG. Такі тести були проведені для запропонованих конфігурацій FiRO-GaRO (тобто 1FiRO-1GaRO, 2FiRO-2GaRO та 4FiRO-4GaRO) на частотах 50 МГц та 100 МГц. Для кожної конфігурації було зібрано та протестовано загалом 320 послідовностей довжиною 1 Гбіт. Параметри кожного тесту були встановлені відповідно до рекомендацій NIST [37]. У Таблиці 6 показані результати, отримані при частоті зразків 50 МГц, тоді як у Таблиці 7 ми презентуємо результати при частоті зразків 100 МГц. Обидва тести відносяться до реалізації вказаної FPGA Stratix IV. Статистичне p-значення повинно бути більшим за 0,01, якщо тест пройдено, і відношення пройдених послідовностей до загальної кількості повинно бути більшим за 0,980. Усі тести вважалися пройденими, оскільки вони мали значення вище порогу. Рівень ентропії виявився подібним у всіх випадках, проте конфігурація 1FiRO-1GaRO досягла трохи нижчих рівнів ентропії.

Test Name	1FiRO-1GaRO		2FiRO-2GaRO		4FiRO-4GaRO	
	p-Value	Proportion	*p*-Value	Proportion	*p*-Value	Proportion
Frequency	0.578763	0.987	0.959132	0.993	0.676097	0.984
BlockFrequency	0.701879	0.987	0.880335	0.990	0.540457	0.987
CumulativeSums *	0.392456	0.990	0.637119	0.987	0.001732	0.987
Runs	0.656634	0.996	0.141256	0.984	0.103676	0.996
LongestRun	0.656634	0.993	0.360699	0.990	0.005789	0.993
Rank	0.585209	0.987	0.839722	0.996	0.202944	0.981
FFT	0.794626	0.978	0.171276	0.990	0.930752	0.993
NonOverlappingTemplate *	0.515367	0.978	0.752361	0.978	0.019520	0.975
OverlappingTemplate	0.280017	0.990	0.124566	0.990	0.727346	0.981
Universal	0.371101	0.981	0.437274	0.984	0.875539	0.978
ApproximateEntropy	0.800471	0.993	0.103676	0.996	0.817667	0.990
RandomExcursions *	0.590375	0.980	0.063657	0.974	0.522989	0.984
RandomExcursionsVariant *	0.652733	0.976	0.153309	0.969	0.511916	0.973
Serial *	0.817667	0.978	0.017156	0.993	0.408942	0.984
LinearComplexity	0.553147	0.993	0.758528	0.993	0.656634	0.981

Таблиця 6. Результати статистичних тестів NIST 800.22 при частоті зразків 50 МГц. * Найгірший випадок, звітований для тестів із кількома результатами.

Test Name	1FiRO-1GaRO		2FiRO-2GaRO		4FiRO-4GaRO	
	p-Value	Proportion	*p*-Value	Proportion	*p*-Value	Proportion
Frequency	0.794626	0.996	0.739918	0.996	0.034455	0.993
BlockFrequency	0.103676	1	0.320988	0.993	0.199580	0.987
CumulativeSums *	0.880335	0.996	0.701879	0.990	0.355569	0.993
Runs	0.971267	0.981	0.695458	0.987	0.109597	0.990
LongestRun	0.365877	0.981	0.990440	0.987	0.708280	0.987
Rank	0.490727	0.990	0.011333	0.993	0.081137	0.993
FFT	0.880335	0.981	0.235285	0.990	0.235285	0.987
NonOverlappingTemplate *	0.969045	0.978	0.177264	0.978	0.392456	0.978
OverlappingTemplate	0.746157	0.990	0.250878	0.981	0.460664	0.987
Universal	0.288780	0.978	0.048716	0.975	0.414525	0.993
ApproximateEntropy	0.521600	0.993	0.371101	0.993	0.213309	0.990
RandomExcursions *	0.120558	0.985	0.825505	0.980	0.247472	0.985
RandomExcursionsVariant *	0.673507	0.980	0.334538	0.990	0.144153	0.985
Serial *	0.546791	0.990	0.297739	0.990	0.202944	0.981
LinearComplexity	0.502986	0.984	0.148968	0.987	0.159799	0.990

Таблиця 7. Результати статистичних тестів NIST 800.22 при частоті зразків 100 МГц. * Найгірший випадок, звітований для тестів із кількома результатами

У цьому розділі був представлений аналіз існуючих Генераторів Випадкових Чисел на Основі Хаотичних Генераторів і методологія для оцінки, який може бути визнаний найкращим варіантом з точки зору ентропії та стійкості до невідповідності місцезнаходження. Ми проілюстрували потік розробки такого ГВЧ для ПЛІС, від визначення архітектури до реалізації, тестування та оцінки продуктивності. Фаза тестування була спрямована на ентропійне джерело, оцінюючи різні сучасні ланцюги ГВЧ на ПЛІС технології та тестуючи їх якість виходу. Ми запропонували конструкцію дуже стійкого ентропійного джерела, застосовуючи кілька FiROs та GaROs,

які показали найкращі показники серед всіх протестованих елементів з точки зору ентропії на ПЛІС пристроях. Архітектура була перевірена за допомогою Набору Оцінки Ентропії NIST на платформі ПЛІС, з оціненою мінімальною ентропією, приблизно рівною 0,995 біта на вихідний біт, відповідно до результатів Така величина довела свою здатність майже не коливатися при різних частотах зразків та конфігураціях. Конфігурація ентропійного джерела вибирається на етапі синтезу, згідно з потребами користувача щодо затримки та області за допомогою конкретних параметрів, які контролюють кількість етапів, а також кількість генераторів на кожному етапі. Ми проілюстрували, як апаратні параметри (кількість етапів та склад кожного етапу) впливають на пропускну здатність та стійкість, не значно впливаючи на ентропію. Зокрема, ми досягли загальної пропускної здатності 400 Мбіт / с для FiGaRO осцилятора з чотирма етапами, 100 Мбіт / с на етап, що відповідає сучасному стану справ у сфері ентропії та складності, а також є покращенням з точки зору пропускної здатності. Відсутність особливих обмежень щодо місцезнаходження та маршрутизації гарантує широкий спектр цільових технологій або різних умов експлуатації.

ВИСНОВКИ

1. Ефективність Методу Фібоначчі: Дослідження показали, що використання Методу Фібоначчі для генерації псевдовипадкових послідовностей на базі пристроїв DDS має великий потенціал у створенні високоякісних випадкових послідовностей.

2. Покращення статистичних характеристик: Оптимізація методів генерації чисел дозволяє покращити статистичні характеристики вихідних послідовностей, що важливо для багатьох сфер технології та індустрії.

3. Важливість практичних застосувань: Результати цих досліджень мають велике значення для практичного використання в розробці технологічних рішень, особливо у сферах телекомунікацій, комп'ютерної техніки та кібербезпеки.

4. Внесок у розвиток сучасних технологій: Вивчення методів генерації псевдовипадкових чисел для пристроїв DDS сприяє підвищенню ефективності сучасних технологій.

5. Потреба в подальших дослідженнях: Незважаючи на досягнуті успіхи, існує потреба в подальших дослідженнях та оптимізації методів генерації випадкових послідовностей для забезпечення їхньої високої якості та надійності.

6. Потенціал для розширення застосувань: Отримані результати можуть мати широкі практичні застосування у багатьох сферах, зокрема в криптографії, мережах зв'язку, а також у розробці нових цифрових технологій.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Jan Pelzl, C.P. Understanding Cryptography; Springer: Berlin/Heidelberg, Germany, 2011Daniel Guijo. Quantum artificial vision for defect detection in manufacturing.

2. Hong, S.L.; Liu, C. Sensor-based random number generator seeding. IEEE Access 2015, 3, 562–568.

3. Rostami, M.; Koushanfar, F.; Karri, R. A Primer on Hardware Security: Models, Methods, and Metrics. Proc. IEEE 2014

4. Baldanzi, L.; Crocetti, L.; Falaschi, F.; Bertolucci, M.; Belli, J.; Fanucci, L.; Saponara, S. Cryptographically secure pseudo-random number generator IP-core based on SHA2 algorithm. Sensors 2020, 20, 1869.

5. Analog and Mixed-Signal Test Methods Using On-Chip Embedded Test Cores: [Електронний ресурс]. — Режим доступу: http://digitool.library.mcgill.ca/webclient/StreamGate?folder_id=0&dvs=1524785121425~634 — (17.11.2017).

6. Characterization of a Pseudo-Random Testing Technique for Analog and Mixed-Signal Built-in-Self-Test: [Електронний ресурс]. — Режим доступу: https://www.researchgate.net/publication/221203272_Characterization_of_a_Pseudo-Random_Testing_Technique_for_Analog_and_Mixed-Signal_Built-in-Self-Test — (12.12.2017).

7.Зсувні регістри з лінійної зворотним зв'язком: [Електронний ресурс]. — Режим доступу: http://um.co.ua/14/14-7/14-73899.html — (06.04. 2018).

8.Алгоритмічні методи генерації псевдовипадкових чисел за рівномірним законом розподілу [Електронний ресурс]. — Режим доступу: https://knhelp.wordpress.com/2012/05/03/лаб-4-алгоритмічні-методигенерації-пс/ — (09.04. 2018).

9.The Marsaglia random number CDROM including the DIEHARD battery of tests of randomness web site: [Електронний ресурс]. — Режим доступу: http://stat.fsu.edu/pub/diehard — (05.03.2018).

10. Orue A.B. Trifork, a new pseudorandom number generator based on lagged Fibonacci maps / A.B. Orue, F. Montoya, L. Hernández Encinas // Journal of computer science and engineering. – 2010. – volume 2, issue 2. – P. 46-51.

11. Burns P. Lagged, Fibonacci Random Number Generators. [Електронний ресурс] // – Режим доступу: http://lamar.colostate.edu/~grad511/lfg.pdf (09.05.2014).

12. Mascagni M. Parallel Pseudorandom Number Generation / M. Mascagni // Advanced architecture Computers. – P. 42-48.

13. Mascagni M. Parallel pseudorandom number generation using additive lagged-Fibonacci recursions / M.Mascagni, M.L. Robinson, D.V. Pryor, S.A. Cuccaro // Springer-Verlag Lecture Notes in Statistics. – 1995. – №106. – P. 263–277.пособие / М.А. Иванов, И.В. Чугунков. – М.: Изд-во НИЯУ МИФИ, 2012. – 400 с.

14. NIST SP 800-22. A Statistical Test Suite for Random and Pseudorandom Number Generators forCryptographic Applications [Електронний ресурс]. Режим доступу:http://csrc.nist.gov/publications/nistpubs//SP80022rev1a.pdf. (12.05.2014).

15. Мандрона М.М. Дослідження впливу параметрів генератора Голлманна на статистичні характеристики вихідного сигналу / Мандрона М.М., Максимович В.М., Костів Ю.М., Гарасимчук О.І. //Вісник кременчуцького національного університету ім. М. Остроградського. – Кременчук: КрНУ, 2013. –Вип. 4 (81). – С. 98-103.

16. Introduction to Cisco IOS Netflow:A Technical Overview / URL: http://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/iosnetflow/prod_white_paper0900aecd80406232.html – свободный. – Загл. С экрана. – Яз. Англ. Дата обращения: 16.03.2017 г.

17. Cisco IOS Flexible NetFlow [Электронный ресурс] / URL: http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/fnetflow/configuration/15-mt/fnf-15-mt-book/fnf-fnetflow.html – свободный. – Загл. с экрана. – Яз. Англ. Дата обращения: 16.03.2017 г.

18. Farkas, P., Rakus, M. Decoding five times extended reed solomon codes using syndromes. Computing and Informatics, 2020, no. 6, vol 39, pp.1311–1335. ISSN 2585-8807 (online).

19. Sowmya, G., Keerthi, K., Lalitkrushna, J. T., et al. An architecture for efficient encoding of quasi cyclic LDPC codes and its implementation in FPGA. In IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT), Indore (India), 2022, pp. 136-140.

20. Liu, J., Feng, Q. A Miniaturized LDPC Encoder: Two-layer architecture for CCSDS near-earth standard. IEEE Transactions on Circuits and Systems II: Express Briefs, 2021, vol. 68, no. 7, p. 2384–2388.

21. Kang, J., Wang, B., Zhang, Y., An, J. Enhanced partially-parallel LDPC decoder for near earth applications. In IEEE 11th International Conference on Electronics Information and Emergency Communication (ICEIEC), Beijing (China), 2021, p. 12-16.

22. Rajagopalan, P., et al. Performance analysis of LDPC decoding algorithms for CCSDS telecommand space data link protocol. In 2nd International Conference for Emerging Technology (INCET), Belagavi (India), 2021, p. 1-5.

23. Rakus, M., Farkas, P. On possible energy savings with transmission supported via feedback channel in CubeSat transceiver. In Journal of Electrical Engineering, 2021, vol.72, no. 5, p. 337–342.

24. Geiselhart, M., Ebada, M., Elkelesh, A. et al., Automorphism ensemble decoding of quasi-cyclic LDPC codes by breaking graph symmetries. IEEE Communications Letters, 2022, vol. 26, no. 8, p. 1705-1709.

25. Nguyen, J., Wang, L., Hulse, C. et al., Neural normalized min-sum message-passing vs. viterbi decoding for the CCSDS line product code. In ICC 2022 - IEEE International Conference on Communications. Seoul (Korea), 2022, p. 2375–2380.

26 Farkas, P., Rakus, M. Adding RLL properties to four CCSDS LDPC codes withouth increasing their redundancy. Accepted for publication in Computing and Informatics, 2023, ISSN 1335-9150

1

Хмельницький національний університет
Факультет інформаційних технологій
Кафедра телекомунікацій, медійних та інтелектуальних технологій

Дипломна робота

Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS

Спеціальність 172 – Телекомунікації та радіотехніка
Виконав: Казіонов Н.А., гр. ТРм-22-1
Керівник – д.т.н., проф. Петрушак В.С.

Мета і задачі дослідження.

2

Метою даної роботи є дослідження ефективності методу Фібоначчі для генерації псевдовипадкових послідовностей у системі прямої цифрової синтезу (DDS). Робота спрямована на:

1. Аналіз можливостей методу Фібоначчі у контексті його використання в системах DDS для створення послідовностей з високою випадковістю.
2. Експериментальне дослідження різних конфігурацій та параметрів методу Фібоначчі для оцінки їхнього впливу на якість та властивості псевдовипадкових послідовностей.
3. Визначення оптимальних умов та параметрів роботи методу Фібоначчі для забезпечення оптимальної випадковості та стабільності отриманих послідовностей у системі DDS.
4. Порівняння результатів використання методу Фібоначчі з іншими алгоритмами генерації псевдовипадкових послідовностей у системах DDS.
5. Визначення можливих обмежень та недоліків методу Фібоначчі в контексті його використання у системах прямої цифрової синтезу.
6. Розробка рекомендацій та висновків щодо оптимального застосування методу Фібоначчі для створення псевдовипадкових послідовностей у системах DDS.

Предмет дослідження полягає в оцінці та аналізі ефективності методу Фібоначчі для генерації псевдовипадкових послідовностей на основі платформи Direct Digital Synthesis (DDS).
Об'єкт дослідження – процес дослідження методу Фібоначчі, який використовується для створення псевдовипадкових послідовностей, особливо в контексті його застосування на базі системи Direct Digital Synthesis (DDS).

Наукова новизна одержаних результатів :

3

Наукова новизна даної роботи полягає в наступному
1. Застосування методу Фібоначчі в DDS: Дослідження використання методу Фібоначчі для створення псевдовипадкових послідовностей у пристроях DDS, що може представляти новий підхід у використанні цього методу.
2. Оцінка якості псевдовипадкових послідовностей: Аналіз характеристик і якості згенерованих послідовностей з використанням методу Фібоначчі в контексті DDS порівняно з іншими відомими методами генерації випадкових чисел.
3. Моделювання та аналіз результатів: Розробка математичних моделей та використання емпіричних методів для оцінки та передбачення властивостей псевдовипадкових послідовностей, згенерованих за допомогою методу Фібоначчі в системах DDS.
4. Оптимізація параметрів методу Фібоначчі: Виявлення оптимальних налаштувань параметрів алгоритму Фібоначчі для отримання високоякісних псевдовипадкових послідовностей у пристроях DDS.
5. Перспективність в практичному застосуванні: Виявлення можливостей використання методу Фібоначчі у пристроях DDS як потенційно ефективного і простого з точки зору апаратної реалізації способу формування псевдовипадкових послідовностей.
6. Аналіз нових вимог та характеристик DDS: Дослідження відповідності згенерованих послідовностей вимогам та характеристикам, що вимагаються у високотехнологічних пристроях DDS.випадкових послідовностей у системах DDS.

Практичне значення одержаних результатів:

4

Отримані результати можуть мати значний практичний вплив:

1. Підвищення ефективності пристроїв DDS: Ці дослідження можуть сприяти поліпшенню якості генерованих псевдовипадкових послідовностей, що в свою чергу позитивно вплине на ефективність та точність пристроїв DDS.

2. Розробка нових технологій індустрії: Результати цього дослідження можуть бути використані в розробці нових технологій, особливо в області цифрових сигнальних систем, комп'ютерної архітектури, телекомунікацій, а також у різних областях індустрії.

3. Підвищення надійності систем: Оптимізовані методи генерації випадкових чисел можуть бути використані для покращення надійності систем у багатьох галузях, таких як криптографія, мережі зв'язку та інші.

4. Застосування у наукових дослідженнях: Отримані результати можуть знайти застосування у наукових дослідженнях, де потрібні випадкові послідовності для моделювання різних фізичних процесів та явищ.

Публікації. Результати дипломної роботи магістра опубліковані в одній статті у науковому журналі «Вимірювальна та обчислювальна техніка в технологічних процесах».

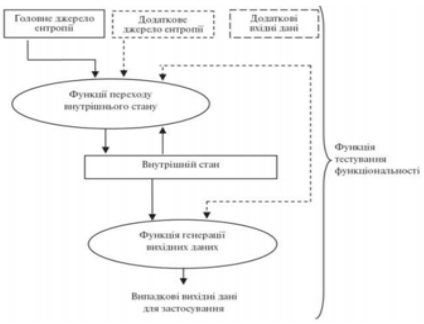


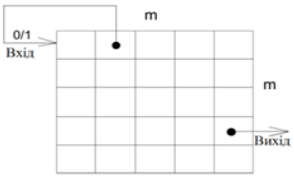
Рисунок 1.3 - Структура генератора випадкових послідовностей

5

Лінійний зсувний регістр зі зворотним зв'язком (LFSR)

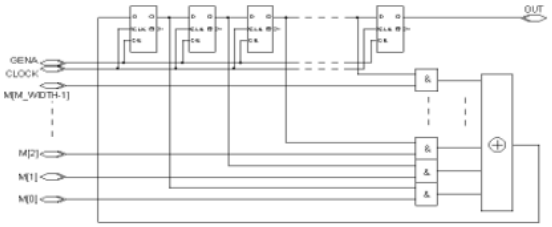
$$degF_1(x) = degF_2(x) = m$$

Рисунок 1.6 - Стовбець матриці генератора



Самоврядний 2-лінійний регістр

Рисунок 1.7 - Рядок матриці генератора



Функціональна схема генератора М-послідовності.

x_n = (a * x_(n-1) + c) % m

де:

x_n - наступний біт послідовності

x_(n-1) - попередній біт послідовності

a - коефіцієнт лінійного рекурентної рівняння

c - константа лінійного рекурентної рівняння

m - модуль лінійного рекурентної рівняння

6

$$T_1 = \begin{vmatrix} a_1 & a_2 & \dots & a_{N-1} & a_N \\ 1 & 0 & & 0 & 0 \\ 0 & 1 & & 0 & 0 \\ & & & & \\ 0 & 0 & \dots & 1 & 0 \end{vmatrix} \text{ або } T_2 = \begin{vmatrix} 0 & \dots & 0 & 0 & a_N \\ 1 & \dots & 0 & 0 & a_{N-1} \\ & & & & \\ 0 & \dots & 1 & 0 & a_2 \\ 0 & \dots & 0 & 1 & a_1 \end{vmatrix}$$

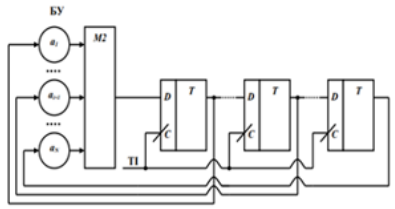


Рис. 1. Схема генератора при k = 1 і T = T1

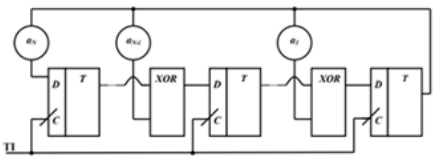
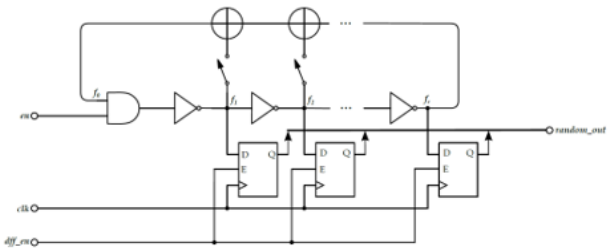
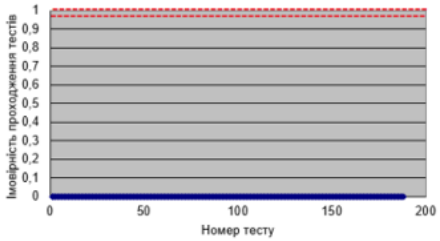


Рис. 2. Схема генератора при k=1 і T= T2

Генератор формування псевдовипадкової послідовності з використанням методу Фібоначчі.

7





Статистичний портрет класичного генератора Фібоначчі

Функціональна схема генератора псевдовипадкової послідовності на основі методу Фібоначчі.

Послідовність Фібоначчі:

Це послідовність чисел, яка визначається наступним рівнянням[2]:

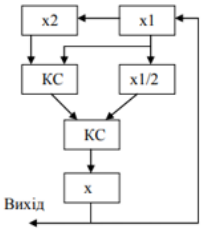
$$f\_n = f\_\{n-1\} + f\_\{n-2\}$$

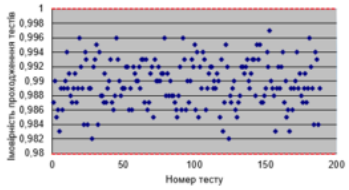
де:

f_n - n-й член послідовності Фібоначчі

f_{n-1} - (n-1)-й член послідовності Фібоначчі

f_{n-2} - (n-2)-й член послідовності Фібоначчі

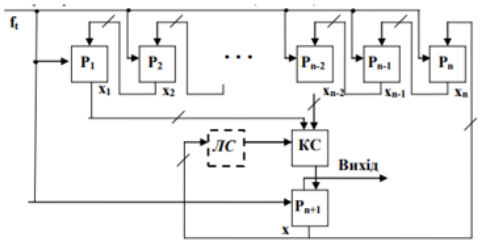




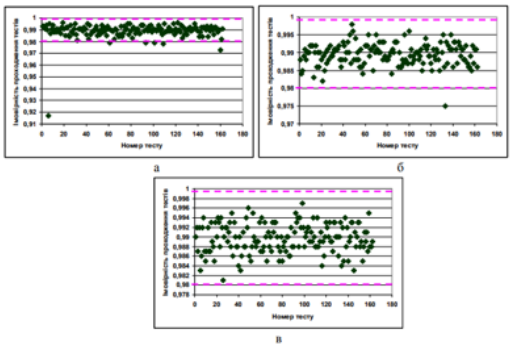
Структурна схема модифікованого генератора Фібоначчі

МОДИФІКАЦІЯ ГЕНЕРАТОРА ФІБОНАЧЧІ

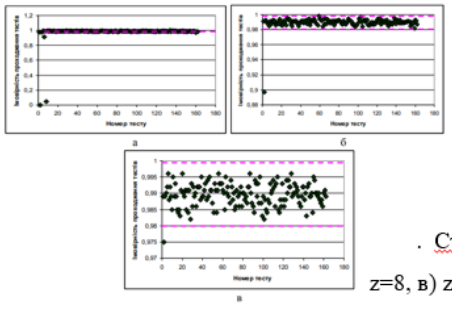
8



Загальна схема модифікованого адитивного генератора Фібоначчі з запізненням

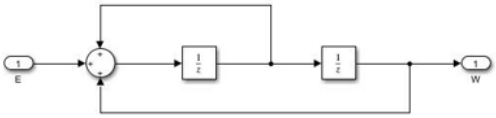


Статистичні портрети МАГФЗ варіант-1: а) z=4, б) z=8, в) z=10

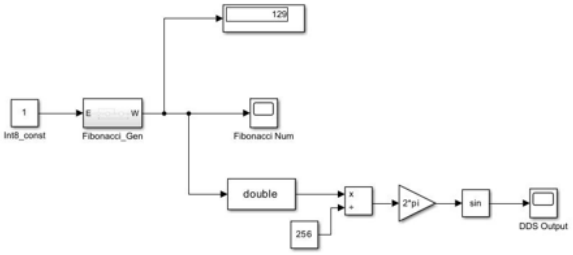


. Статистичні портрети МАГФЗ варіант-2: а) z=4, б) z=8, в) z=10

РОЗРОБКА ТА ДОСЛІДЖЕННЯ ФОРМУВАЧА ПВП НА БАЗІ DDS 9



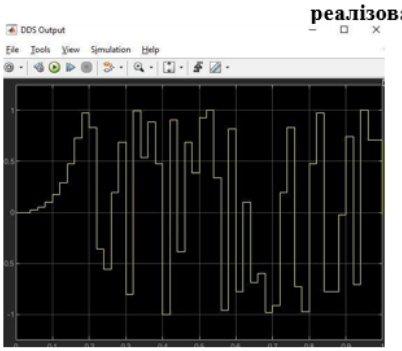
Внутрішня структура моделі.



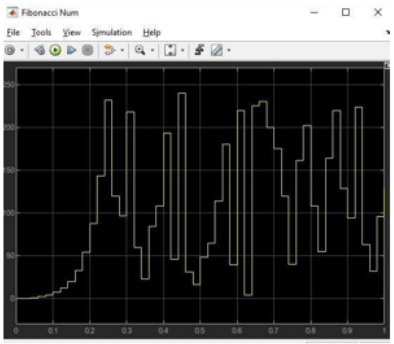
Модель яка реалізує генерацію чисел Фібоначчі через 2 елементи затримки та сумматор.

Формат данних стоїть int8, тому значення скидується при переповненні. Вихідний сигнал з системи конвертується в формат double і передається функції sinus (як в dds але без ЦАП на виході). Звісно це просто модель.

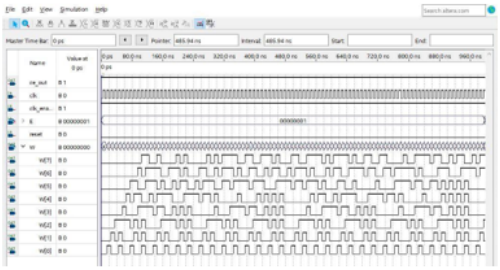
Дослідження характеристик формувача ПВП на базі DDS реалізованого на ПЛІС. 10



Графік чисел Фібоначчі.



Графік чисел Фібоначчі на виході функції sin.



В програмному середовищі Quartus було побудована наша модель після якої ми отримали наступні результати .

Входу E - enable треба дати якесь значення відмінне від нуля інакше не запрацює (в сімулінку це константа 1 на вхід)

W - це 8 бітне число фібоначі

Висновки

1. Ефективність Методу Фібоначчі: Дослідження показали, що використання Методу Фібоначчі для генерації псевдовипадкових послідовностей на базі пристроїв DDS має великий потенціал у створенні високоякісних випадкових послідовностей.

2. Покращення статистичних характеристик: Оптимізація методів генерації чисел дозволяє покращити статистичні характеристики вихідних послідовностей, що важливо для багатьох сфер технології та індустрії.

3. Важливість практичних застосувань: Результати цих досліджень мають велике значення для практичного використання в розробці технологічних рішень, особливо у сферах телекомунікацій, комп'ютерної техніки та кібербезпеки.

4. Внесок у розвиток сучасних технологій: Вивчення методів генерації псевдовипадкових чисел для пристроїв DDS сприяє підвищенню ефективності сучасних технологій.

5. Потреба в подальших дослідженнях: Незважаючи на досягнуті успіхи, існує потреба в подальших дослідженнях та оптимізації методів генерації випадкових послідовностей для забезпечення їхньої високої якості та надійності.

6. Потенціал для розширення застосувань: Отримані результати можуть мати широкі практичні застосування у багатьох сферах, зокрема в криптографії, мережах зв'язку, а також у розробці нових цифрових технологій.

Довідка: ВХНУ ТН 22/11/23

Видання: Вісник Хмельницького національного університету. Технічні науки

Категорія фаховості видання: фахове видання України, у якому можуть публікуватися результати дисертаційних робіт на здобуття наукових ступенів доктора наук, кандидата наук та ступеня доктора філософії, категорії «Б» філософії, категорії «Б» (наказ МОН №1643 від 28.12.2019, наказ МОН №409 від 17.03.2020).

Напрям – технічні науки за спеціальностями – 101, 121, 122, 123, 124, 125, 141, 151, 161, 172, 181, 182 (28.12.2019), спеціальності – 131, 132, 133 (17.03.2020)

Назва статті: ОГЛЯД АПАРАТНИХ ЗАСОБІВ ФОРМУВАННЯ ПСЕВДОВИПАДКОВОЇ ПОСЛІДОВНОСТІ

Автори: ПЕТРУШАК В.С., КАЗІОНОВ Н. А. (Хмельницький національний університет)

Номер, у який прийнято статтю: №6 до друку рекомендовано буде до 25 грудня 2023 року.

22.11.2023

Начальник відділу
інтелектуальної власності та трансферу технологій Ю.В.Кравчик

ВІДДІЛ КАДРІВ І.С.Мартинюк

Sun Dec 17 20:14:21 EET 2023, Стецюк Віктор Іванович, Хмельницький національний університет, ХН

Anti-Plagiarism v-15.257

Максимальное совпадение с одним документом 0.0%

Словари проверки: en_US, ru_RU, ua_UA. **Ошибок в документах: 9%**

	Документ		Суммарное совпадение по Базе Данных	
ID: 123580 Название: Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS Добавлено в БД: 2023-12-17 Авторы: Казіонов Нікіта Андрійович Руководители: Петрушак Володимир Степанович Консультанты: Опоненты:	Символы	Лексемы	Символы	Лексемы
	98185	1555	826 (1%)	16 (1%)

Источник плагиата

ID	Описание	Наличие плагиата в документе	
		Символы	Лексемы

UNICHECK
by Turnitin

Имя пользователя:
Kafedra TMIT KhNU

ID проверки:
1016015156

Дата проверки:
17.12.2023 22:27:26 EET

Тип проверки:
Doc vs Internet + Library

Дата отчета:
17.12.2023 22:28:13 EET

ID пользователя:
100005657

Название файла: Казіонов_ТРм_22

Количество страниц: 108 Количество слов: 15712 Количество символов: 123161 Размер файла: 2.33 MB ID файла: 1015701816

6.43%
Совпадения

Наибольшее совпадение: 2.49% с Интернет-источником (https://ela.kpi.ua/bitstream/123456789/23213/1/TkachenkoMH.

6.33% Источники из Интернета 185 ..... Страница 110

0.59% Источники из Библиотеки 60 ..... Страница 111

0% Цитат

Не найдено ни одной цитаты

Не найдено ни одной ссылки

0% Исключений

Нет исключенных источников

Модификации

Обнаружены модификации текста. Подробная информация доступна в онлайн-отчете.

Замененные символы 17

В І Д Г У К

на дипломну роботу студента групи ТРм-22-1

Казіонова Нікіти Андрійовича

«Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS»

Отримав подальший розвиток метод Фібоначчі для формування псевдовипадкової послідовності на базі цифрового прямого синтезатора частоти. Це створило новий підхід в галузі формування псевдовипадкової послідовності і надає можливості для подальших досліджень в цьому плані.

Під час виконання кваліфікаційного проекту студент Казіонов Н. А., з належною наполегливістю віднісся до вирішення поставлених завдань, зарекомендував себе кваліфікованим спеціалістом в області телекомунікацій та радіотехніки з глибокими системними теоретичними знаннями та добрими практичними навичками.

В цілому дипломна робота Казіонова Н. А., “Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS” відповідає вимогам до дипломних робіт магістерського рівня, заслуговує на оцінку “добре”, а її автор – на присвоєння кваліфікаційного рівня магістер зі спеціальності 172 – “Телекомунікації та радіотехніка”.

Науковий керівник _____ (Потрусов В. С.)

“17” 12 2023 р.

РЕЦЕНЗІЯ

на дипломну роботу студента групи ТРм-22-1

Казіонова Нікіти Андрійовича

«Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS»

Кваліфікаційна робота магістра присвячена питанням розробки системи прихованого зв'язку із застосуванням хаотичних принципів обробки сигналів та аналізу можливостей її використання за умови суттєвих нелінійностей в каналі передачі.

Робота складається із вступу, 4 розділів, висновків по роботі, переліку із 27 джерел посилання та 2 додатків (14 сторінок). Загальний обсяг роботи в якому викладено основний зміст складає 107 сторінок і містить 25 рисунків на 87 сторінках по тексту . Повний обсяг роботи - 108 сторінок.

У вступі підкреслюється актуальність тематики роботи для майбутніх досліджень генераторів частоти , та методів формування псевдовипадкової послідовності. Визначено мету, завдання, наукову новизну та практичну цінність роботи.

Перший розділ роботи присвячено огляду основних способів формування випадковостей і їх огляд.

Другий розділ присвячено розгляду основних варіантів генераторів формування випадковостей і їх детальний аналіз та статистичні тести.

В третьому розділі йде дослідження моделі і функціональних блоків для розробки найкращої по характеристикам моделі.

Четвертий розділ охоплює синтез імітаційної моделі та окремих її субмодулів в середовищі Matlab\Simulink. Розроблені критерії оцінювання якості передачі, методики вибору вхідних параметрів та методики дослідження. Проведено імітаційне моделювання, отримані нові результати та рекомендації до використання в системах прихованого зв'язку.

Процес висвітлення головних позицій щодо завдань досліджень є логічно пов'язаним. Наведені у роботі припущення мають достатнє обґрунтування. Оформлення пояснювальної записки знаходиться на рівні, що відповідає стандарту університету.

Серед позитивних сторін магістерської роботи слід відмітити наступне:

- актуальність тематики для майбутніх проєктів;
- запропоновані критерії та методики формування випадковості є доцільними;
- отримані результати підтверджують загальні теоретичні припущення;
- розроблена імітаційна модель може використовуватись для подальших досліджень даної галузі.

Серед недоліків слід відмітити:

- деякі пункти тексту мають слабке відношення до тематики досліджень;
- частина нетрадиційних скорочень застосована для традиційних термінів;
- результати моделювання отримані лише для однієї групи вхідних параметрів системи, варто було провести ширше моделювання із представленням результатів.

Загалом дипломна робота магістра Казіонова Нікіти Андрійовича є актуальною, під час роботи отримані вагомі результати, робота заслуговує на оцінку "відмінно".

Рецензент: Зав. каф. АКІТ та Р
д.т.н., проф. Мартенюк В.В.

Завідувачу кафедри
телекомунікацій, медійних та
інтелектуальних технологій (ТМІТ)
Підченко С.К.
здобувача вищої студента
2 курсу, гр. ТРм-22-1
Казіонова Нікіти Андрійовича

ЗАЯВА

З правилами чинного Положення «Про дотримання академічної доброчесності в Хмельницькому національному університеті» від 26.09.2020 (зі змінами від 26.11.2020), згідно з яким виявлення плагіату є підставою для відмови в допуску кваліфікаційної роботи до захисту та застосування заходів дисциплінарної та академічної відповідальності, ознайомлений. Про використання програмно-технічних засобів для перевірки кваліфікаційних робіт здобувачів вищої освіти на плагіат оповіщена та надаю свою згоду на обробку та збереження університетом моєї роботи в інституційному репозитарії університету.

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

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

1.12.23 _____ Казіонов Н.А.

дата підпис

РІШЕННЯ КАФЕДРИ

ТЕЛЕКОМУНІКАЦІЙ, МЕДІЙНИХ ТА ІНТЕЛЕКТУАЛЬНИХ ТЕХНОЛОГІЙ

ПРО ДОПУСК ДИПЛОМНОЇ РОБОТИ ДО ЗАХИСТУ

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

Назва: Метод Фібоначчі для формування псевдовипадкової послідовності на базі DDS

Автор: **Казіонов Нікіта Андрійович**

Спеціальність: **172 Телекомунікації та радіотехніка**

Освітня програма: Телекомунікації та радіотехніка

Науковий керівник: **к.т.н., доц. Петрушак Володимир Степанович**

Після аналізу звіту подібності зроблено такий висновок:

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

Підтвердження: Запозичення у розмірі 6.43%, виявлені в роботі відповідають тексту стандартних бланків, решта запозичень є випадковими, тому ці запозичення не є плагіатом, бо вони не стосуються практичної значущості роботи.

17 . 12 .2023р.

Науковий керівник
к.т.н.,доц. Петрушак В.С.

Відповідальний за перевірку на плагіат
к.т.н.,доц. Пивовар О.С.

Зав.каф. ТМІТ
д.т.н., проф. Підченко С.М.