Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет «Комп'ютерні технології і системи»
(назва факультету)

Кафедра «Електронні обчислювальні машини»
(повна назва кафедри)

26.12.2022

Пояснювальна записка

до кваліфікаційної роботи
магістра
(ступінь вищої освіти)

на тему: Дослідження та розробка апаратно-програмних комплексів засобів генерації випадкових чисел. Комплекс генерації випадкових чисел на базі мікроконтролерів
за освітньою програмою Комп'ютерна інженерія
зі спеціальності: 123 Комп'ютерна інженерія
(шифр і назва спеціальності)

Виконав: студент групи: КС2121

	(підпис студента)	Альбіна МАСЛАК (Ім'я ПРІЗВИЩЕ)
Керівник:	(підпис)	доцент, Денис ОСТАПЕЦЬ (посада, Ім'я ПРІЗВИЩЕ)
Нормоконтролер:	(підпис)	доцент, Володимир ШАПОВАЛОВ (посада, Ім'я ПРІЗВИЩЕ)

Засвідчую, що у цій роботі немає запозичень з праць інших авторів без відповідних посилань.

Студент _____
(підпис)

Дніпро – 2022 рік

Ministry of Education and Science of Ukraine
Ukrainian State University of Science and Technologies

Faculty «Computer technologies and systems»
(faculty)

Department «Electronic computers»
(department)

Explanatory Note
to Master's Thesis

Master's
(higher education degree)

on the topic: Research and development of hardware and software complexes of means for random numbers generating. Random number generating complex based on microcontrollers.

in the Speciality: 123 Computer Engineering
(speciality and its code)

Done by the student of the group: KC2121 Albina Maslak
(name, surname)

Scientific Supervisor: Associate Professor, Denys Ostapets
(position, name, surname)

Normative controller : Associate Professor, Volodymyr Shapovalov
(position, name, surname)

Dnipro – 2022

Міністерство освіти і науки України
Український державний університет науки і технологій

Факультет: Комп'ютерні технології і системи
Кафедра: ЕОМ
Рівень вищої освіти: Другий (магістерський)
Освітня програма: Комп'ютерна інженерія
Спеціальність: 123 Комп'ютерна інженерія
(шифр та назва)

ЗАТВЕРДЖУЮ
Завідувач кафедри ___
_____ _____
(підпис) (Ім'я ПРІЗВИЩЕ)
Дата 11.04.2022

З А В Д А Н Н Я

на кваліфікаційну роботу магістра
(ступінь вищої освіти)

студенту Маслак Альбіні Володимирівні
(Прізвище, Ім'я По батькові)

1. Тема роботи: Дослідження та розробка апаратно-програмних комплексів засобів генерації випадкових чисел. Комплекс генерації випадкових чисел на базі мікроконтролерів

Керівник роботи: Остапець Денис Олександрович, к. т. н., доцент
(Прізвище, Ім'я, По батькові, науковий ступінь, вчене звання)

затверджені наказом від " 29 " 03 2022 р. № 284ст

2. Строк подання студентом роботи: 19.12.2022 р.
3. Вихідні дані до роботи:
- Методи та алгоритми генерації випадкових чисел;
- Методи та засоби перевірки чисел на випадковість.

4. Зміст пояснювальної записки (перелік питань, які потрібно опрацювати):
4.1 Аналітична частина:
- Аналіз характеристик джерел шуму для реалізації апаратного генератора випадкових чисел.

4.2 Основна частина:
- Огляд та аналіз джерел шуму для генерації випадкових чисел;
- Склад, функції та режими роботи комплексу;
- Розробка апаратної частини комплексу;
- Розробка програмного забезпечення серверної та клієнтської частин;
- Дослідження якості отримуваних випадкових чисел.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень):
- Характеристики джерел шуму;
- Організація та функції комплексу;

- Схеми апаратної частини комплексу;
- Основні алгоритми програм;
- Приклади роботи комплексу;
- Результати дослідження якості отриманих випадкових чисел.

5. Консультанти розділів роботи:

Розділ	Прізвище, ініціали та посада консультанта	Завдання видав: (підпис консультанта, дата)	Завдання прийняв: (підпис студента, дата)

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Огляд та аналіз джерел шуму для генерації випадкових чисел	Строк виконання етапів роботи	Примітка
2	Склад, функції та режими роботи комплексу	21.11.2022	20%
3	Розробка апаратної частини комплексу	25.11.2022	15%
4	Розробка програмного забезпечення серверної та клієнтської частин	02.12.2022	20%
5	Дослідження якості отримуваних випадкових чисел	09.12.2022	20%
6	Реферат, вступ, висновки	16.12.2022	20%
7	Подання кваліфікаційної роботи до кафедри	19.12.2022	5%
8	Захист кваліфікаційної роботи на засіданні Екзаменаційної комісії	19.12.2022	

Студент _____ (підпис) Альбіна МАСЛАК (Ім'я ПРІЗВИЩЕ)

Керівник роботи _____ (підпис) Денис ОСТАПЕЦЬ (Ім'я ПРІЗВИЩЕ)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 88с., 38 рис., 5 табл., 11 додатків, 31 джерело.

Об’єктом дослідження є методи та засоби генерації випадкових чисел.

Метою роботи є розробка аппаратно-програмного комплексу генерації випадкових чисел на базі мікроконтролера та оцінка якості отримуваних випадкових чисел.

Методи дослідження – експериментальне дослідження якості отримуваних випадкових чисел з використанням кейсів статистичних та графічних тестів та спеціального програмного забезпечення.

Здійснено огляд та аналіз джерел шуму для генерації випадкових чисел. Обрано у якості джерела шуму лавинний шум. Розроблено склад, функції та режими роботи комплексу. Описано принципи обміну між елементами комплексу та принцип оцифровування шумів. Розроблено програмне забезпечення серверної та клієнтської частин. Досліджено якість отримуваних чисел за допомогою статистичного та графічного тестів.

Ключові слова: МІКРОКОНТРОЛЕР, СТАБІЛІТРОН, ARDUINO, ВИПАДКОВІ ЧИСЛА, ПСЕВДОВИПАДКОВІ ЧИСЛА, ЛАВИННИЙ ШУМ, BBS, ATMEGA 2560, ДЖЕРЕЛА ШУМУ, ГЕНЕРАТОР, TCP, ПОСЛІДОВНИЙ ПОРТ, GOLANG, ARDUINO IDE, СЕРВЕР, КЛІЄНТ.

ЗМІСТ

ВСТУП

На сьогодні проблема генерації випадкових чисел актуальна як ніколи. Для вирішення цієї проблеми було створено багато методів генерації випадкових даних, деякі з яких вже існували дуже давно, серед яких відомі такі, як підкидання кубиків, підкидання монети та інше. Зараз використовують найчастіше два основних методи генерації: програмний та апаратний. Серед них апаратний метод дає можливість отримувати аперіодичні та непередбачувані числа. Випадкові числа є виміряними значеннями будь-якого випадкового фізичного процесу, наприклад як лавинний шум. Повторити аналогічний фізичний процес неможливо, а значить, що ми отримуємо випадкові числа, які не будуть повторюватися

Галузь генерації випадкових чисел на базі мікроконтролера достатньо нова, але ще не до кінця вивчена. Питання генерації дійсно випадкових чисел піднімається щодня для забезпечення безпеки даних в пристроях. Тому тема дипломної роботи є актуальною.

Тема роботи затверджена наказом по університету №284ст від 29.03.2022 р.

Метою роботи є розробка аппаратно-програмного комплексу генерації випадкових чисел на базі мікроконтролера та оцінка якості отримуваних випадкових чисел.

Основні положення роботи доповідались та були схвалені на XVI Міжнародній науково-практичній конференції «Сучасні інформаційні та комунікаційні технології на транспорті, в промисловості та освіті» у 2022 році (див. додаток А), опубліковано тези доповіді [1].

Представлена робота складається зі вступу, 5 розділів та висновків.

У розділі 1 представлений огляд та порівняльний аналіз джерел шуму для генерації випадкових чисел.

У розділі 2 розроблено узагальнену структуру комплексу та принципи взаємодії його елементів між собою.

У розділі 3 представлено розробку апаратної частини комплексу на базі плати Arduino Mega 2560.

У розділі 4 проведена розробка програмного забезпечення серверної та клієнтської частин комплексу.

У розділі 5 наведено результати дослідження якості отримуваних випадкових чисел.

В додатку А наведено копію тез доповіді на конференції, в додатках Б-Л наведено вихідні коди програм, у додатку М наведено згенеровані випадкові числа.

1 ОГЛЯД ТА АНАЛІЗ ДЖЕРЕЛ ШУМУ ДЛЯ ГЕНЕРАЦІЇ ВИПАДКОВИХ ЧИСЕЛ

1.1 Загальні відомості

У наш час досить часто знаходять застосування послідовності чисел, вибраних випадково з деякої множини [2]. Як приклади завдань, у яких використовуються випадкові числа, можна навести такі:

а) тестування алгоритмів;

б) імітаційне моделювання;

в) деякі завдання чисельного аналізу;

г) імітація введення користувача.

Випадкове число (random variate, random number) – це число, що є реалізацією випадкової величини. Зараз у період, коли технології розвиваються з неймовірною швидкістю люди дуже часто використовують комп'ютерні системи для генерації випадкових чисел, але вони малоефективні [3]. Для відеоігор ці функції, можливо, забезпечують достатньо випадковості для певних завдань, але є непридатними в тих випадках, коли потрібна якісна випадковість. Наприклад, у статистиці, криптографічних програмах або у чисельному аналізі. Для отримання випадкових результатів ще здавна люди використовували гральні кості, підкидували монети, тасували ігрові карти та ін.

Зараз для отримання випадкових чисел існує 3 основні підходи [4]:

а) використання таблиць випадкових чисел;

б) генерування випадкових чисел за допомогою деякого алгоритму;

в) використання спеціальних апаратних пристроїв – датчиків випадкових чисел.

Якщо є досить велика таблиця випадкових чисел, то питання про джерело незалежних реалізацій випадкової величини можна вважати принципово вирішеним. На практиці зберігання великої таблиці в пам'яті обчислювальної машини представляється дуже незручно. Виходячи з цього, то використання таблиць випадкових чисел не дуже доречне але поодинокі випадки

використання є. З недоліків можна виділити те, що таблиці існують для обмеженого набору законів розподілу.

Псевдовипадкове число (pseudo-random number) – це число, отримане відповідно із заданим алгоритмом, що використовується як випадкове число. На основі рекурентних співвідношень отримують велику кількість псевдовипадкових послідовностей. Однак, ці послідовності є періодичними, тобто кожне наступне число визначається на підставі певної комбінації попередніх значень. Виходячи з цього після генерації псевдовипадкових чисел послідовності обов’язково будуть періодичними і передбачуваними.

Апаратний генератор випадкових чисел (генератор істинно випадкових чисел) – пристрій, який генерує послідовність випадкових чисел на основі виміряних параметрів з пристроїв, які хаотично видають певні шуми протікаючого фізичного процесу [5]. Генератор містить внутрішнє фізичне джерело шуму та виробляє аналоговий сигнал, який потім дискретизується. Отриманий сигнал перетворюється на випадкову послідовність.

Перевагою апаратних генераторів є те, що випадкові числа є виміряними значеннями будь-якого випадкового фізичного процесу. Тобто значення ми отримуємо аперіодичні і непередбачувані. До недоліків можна віднести неможливість повторного відтворення обчислень. Це загалом логічно, так як повторити аналогічний фізичний процес неможливо. Також є можливість отримувати непередбачувані зміни параметрів пристрою під впливом зовнішніх факторів – температури, електричних та магнітних полів та старіння елементної бази.

На рис. 1.1 представлена схема джерел шумів для генераторів випадкових чисел.

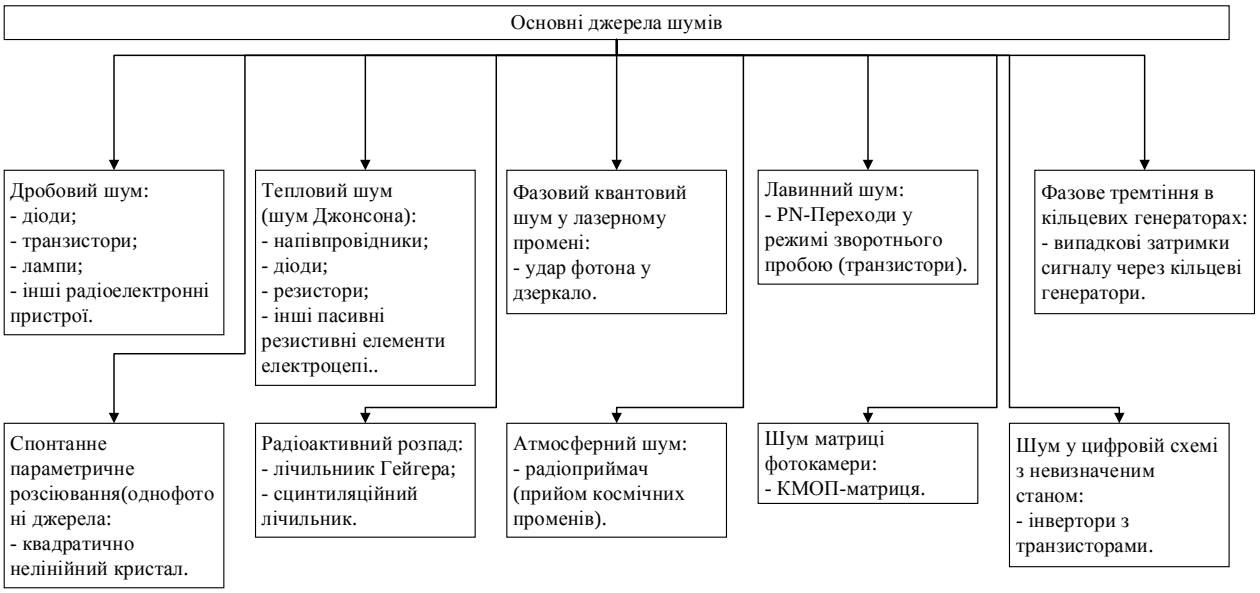

Рисунок 1.1 – Основні джерела шумів

1.2 Джерела шуму для генерації випадкових чисел

1.2.1 Дробовий шум

Дробовий шум – шум в радіоелектронних пристроях, таких як діоди, транзистори та лампи. Цей шум пов’язаний з дискретністю носіїв зарядів, які утворюють струми. Флуктуації виникають внаслідок статистичної незалежності впорядкованого переміщення носіїв зарядів. При електронному, дірковому та іонному струмах утворюються шуми [6].

Характеристики дробового шуму:

а) шум виникає лише при протікання струму;

б) струм дробового шуму не залежить від температури;

в) спектральна щільність дробового шуму залежить від частоти;

г) дробовий шум проявляється при протіканні струму через будь-який провідник.

У провідниках бар'єри утворюються за рахунок домішок та неоднорідностей, що завжди присутні в металах. Рівень дробового шуму при цьому дуже малий через малу відносної величини бар'єрів у провідниках та величезної кількості електронів, що у перебігу струму. У напівпровідниках дробовий шум виражений набагато сильніше [7].

1.2.3 Тепловий шум

Тепловий шум (шум Джонсона) – шум у пасивних елементах схеми, з яких після посилення виходить генератор випадкової напруги. Причина появи шуму – випадковий броунівський рух електронів у резистивному середовищі. Тепловий шум збільшується зі зростанням температури і опору і часто являється найважливішою складовою шуму в напівпровідникових перетворювачах даних [8].

Генератор це один із успішних прикладів побудови генератора випадкових чисел на базі теплового шуму. Цей генератор розроблений компанією Intel у 1999 році та використаний у наборі чипів серії Intel 800. Генератор випадкових чисел Intel використовує послідовності випадкових чисел, які отримані з 2 тактових генераторів, частота роботи одного з яких перевищує частоту іншого у 100 разів.

1.2.4 Фазовий квантовий шум у лазерному промені

Квантовий шум у дзеркалі це один із найнадійніших способів отримання випадкових чисел. На основі цього створено генератор випадкових чисел, що фіксує квантовий ефект удару фотонів у дзеркало. На напівпрозоре дзеркало направляють фотони, які генеруються джерелом одиночних фотонів. Фотон може відбитися від дзеркала, а може пройти через напівпрозоре дзеркало з однаковими частками ймовірності. Саме цей вибір є абсолютно випадковим. На виході системи стоять два лічильники фотонів, що реєструють фотони, які пройшли через дзеркало чи відбилися, та формують вихідні електричні сигнали [8].

На рис 1.2 розділювач променю використовують для випадкових чисел генератора.

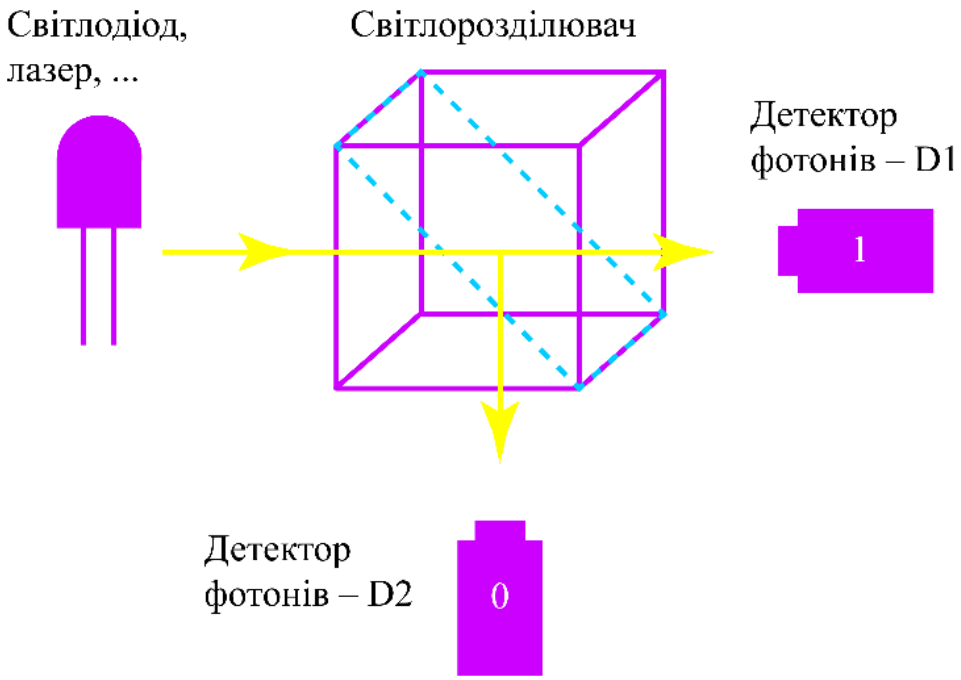

Рисунок 1.2 – Розділювач променя

Далі два детектори фотонів D1 і D2 для виявлення двох можливих результатів. Таким чином, кожен фотон, що потрапляє у розділювач променю, генерує один випадковий фотон, а тобто двійковий розряд [9].

Подібні квантові генератори мають високу швидкість генерації випадкових чисел приблизно 10-16 Мбіт/с, – при якої не спостерігається жодних кореляцій та виконуються всі статистичні тести [10].

1.2.5 Лавинний шум

Джерелами лавинного шуму є PN-переходи, що працюють у режимі зворотного пробою. PN-переходи наявні у стабілітронах (діодах Зенера). Під час лавинного пробою генерується струм, який складається з випадково розподілених шумових викидів, що проходять через зміщений перехід. Лавинний шум дуже схожий на дробовий шум, тому для генерації лавинного шуму потрібна наявність струму. У генераторах випадкових чисел зазвичай використовують перехід емітер-база біполярного NPN-транзистора через низьку пробійну напругу. Знятий з переходу шум посилюється і перетворюється на цифровий сигнал. Випадкові числа з подібних генераторах

проходять усі статистичні тести, проте швидкість їхньої генерації вкрай мала - менше 10 Кбіт/с [8].

Схема генератора лавинного шуму представлена на рис. 1.3 [11].

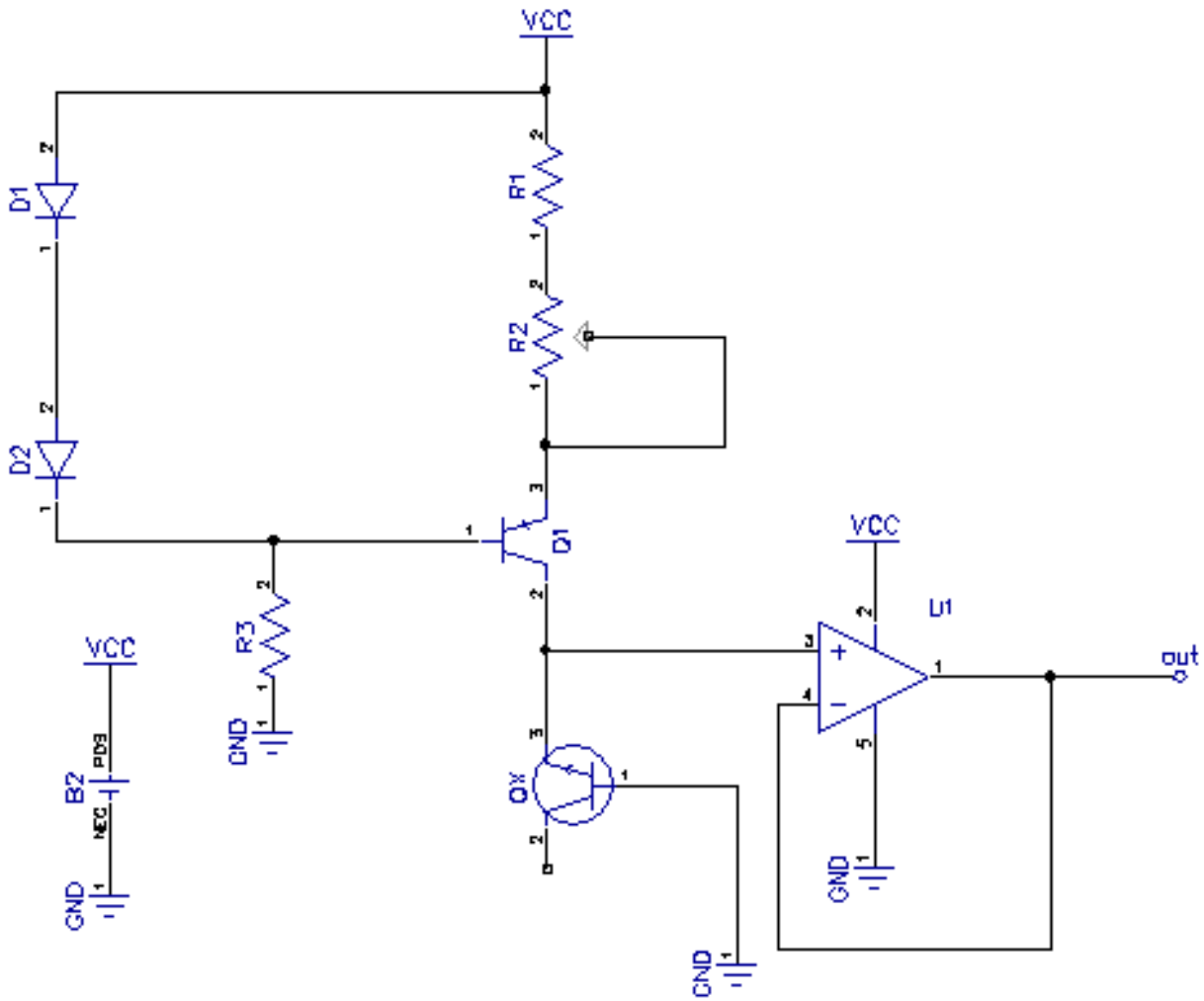

Рисунок 1.3 – Схема генератора лавинного шуму

Схема складається з генератора постійного струму та буфера, а тестовий пристрій позначено як QX. Генератор постійного струму складається з двох діодів D1 і D2, трьох резисторів R1, R2 і R3 і транзистора PNP Q1 [11].

1.2.6 Фазове тремтіння в кільцевих генераторах

Фазове тремтіння у кільцевих осциляторах забезпечує простий і ефективний спосіб вилучення випадковість. Істинний генератор випадкових чисел на основі кільцевих осциляторів зараз дуже популярні, оскільки їх можна ефективно

реалізувати в інтегральній мікросхемі, а також у проєктах на основі ПЛІС. Кільцевий осцилятор побудований за допомогою непарної кількості інверторів і подачі вихідного сигналу кінцевого інвертора на вхід першого інвертора, який представлений на рис.1.4.

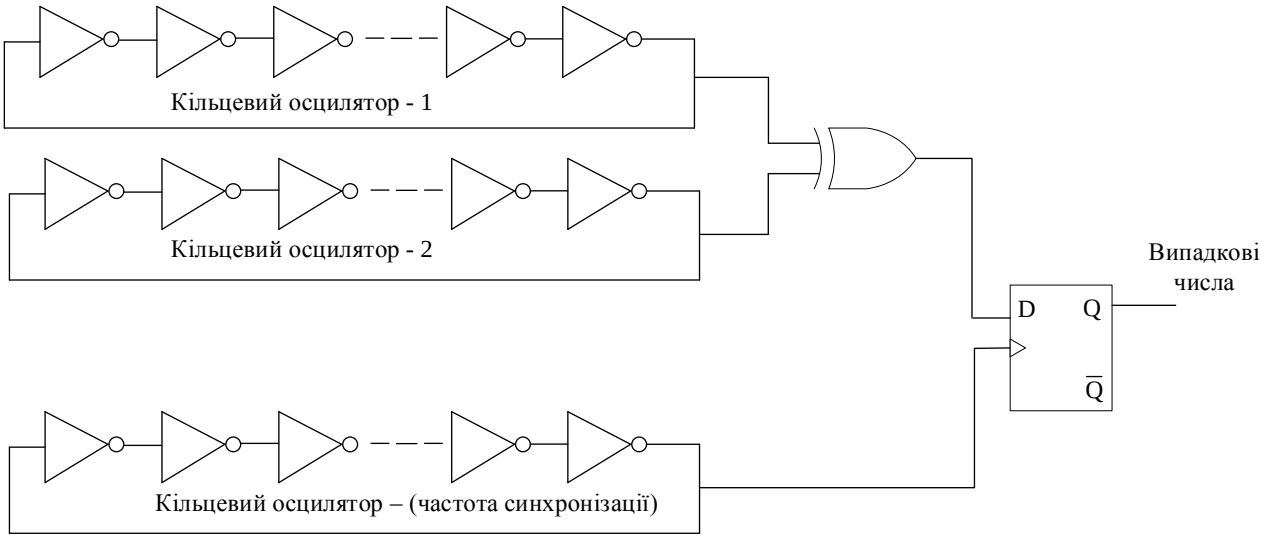

Рисунок 1.4 – Кільцевий генератор на основі істинного генератора випадкових чисел

Зміни в джерелі живлення у вигляді шуму джерела живлення викликають зміни в затримці інвертора і, отже, в частоті коливань. Це невизначеність сигналу осцилятора (вихід) у часовій області називається джиттером. Оскільки тремтіння залежить від різних факторів, то деякі з них є випадковими. Основні схеми істинного генератора випадкових чисел використовують два або більше кільцевих генераторів, XOR виходи. Вихідний сигнал XOR дискретизується під час перехідної зони, щоб стати випадковим значенням через тремтіння в двох кільцевих осциляторах [12].

1.2.7 Спонтанне параметричне розсіювання

Однофотонні джерела є найважливішими квантовими елементами, що використовуються у розподілі квантових ключів, генерації випадкових чисел, квантових обчисленнях та квантової метрології. Тому спонтанне параметричне розсіяння (СПР) також можна використовувати в генераторах випадкових чисел

Одним із найпростіших методів отримання одиночних фотонів є спонтанне параметричне розсіювання світла (СПР) у квадратично-нелінійному кристалі. У процесі СПР фотони поля накачування випадково розпадаються на пари фотонів (один із яких зазвичай називається сигнальним, а інший холостим) згідно із законами збереження енергії та імпульсу [13].

1.2.8 Радіоактивний розпад

Як джерело шуму також можна використовувати радіоактивний розпад, оскільки для нього характерна випадковість кожного окремого акту розпаду. У результаті на приймач (наприклад, лічильник Гейгера або сцинтиляційний лічильник) у різні проміжки часу потрапляє різна кількість частинок [14].

1.2.9 Атмосферний шум

Одним із простих джерел шуму є атмосферний шум з використанням радіоприймача, який знімає дані шумів. Також цим приймачем можна приймати космічні промені, кількість яких у різні проміжки часу випадкова [15].

1.2.10 Матриця фотокамери

Вчені з Женевського університету на чолі з Бруно Сангінетті дослідили, що більшість джерел світла випускають фотони в випадкові моменти часу і кількість фотонів, випущених за одиницю часу буде різнитися на величину, яка є цілком випадковою. Цей факт ліг в основу предметно орієнтованої мови, побудованої на базі світлочутливої КМОН-матриці звичайної.

Кожен піксель матриці рахує кількість фотонів, що потрапили на його поверхню за певний проміжок часу. Ці фотони перетворюються в електрони, які множаться на множник, визначений світлочутливістю матриці (рівень ISO). Кількість електронів за той самий період буде відрізнятися на цілком випадкове число [8]. Схема генератора випадкових чисел зображено на рис. 1.5.

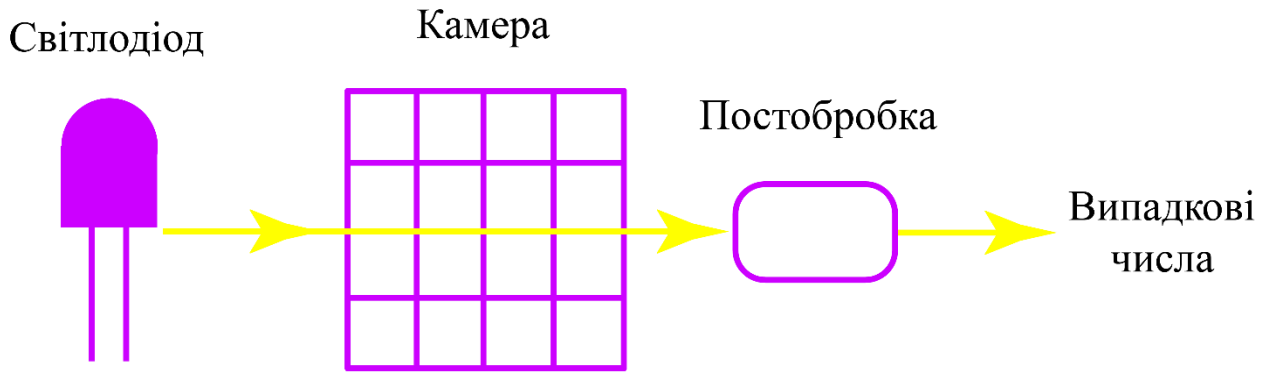

Рисунок 1.5 - Схема генератора випадкових чисел.

Необроблене двійкове представлення піксельних значень об’єднується та передається через екстрактор випадковості. Цей екстрактор виводить квантовий випадковий результат чисел.

1.2.11 Цифрова схема з невизначеним станом

Компанія Intel у 2008 році вирішила розробити новий пристрій з двома логічними станами (низький та високий), хоча до 2008 року це рішення ще ніхто не використовував. І завжди схеми працювали виключно в одному із станів.

Схема генератора випадкових чисел складається з кількох інверторів, вихід кожного з яких підключений до входу іншого. Якщо на виході першого інвертора буде логічний низький рівень сигналу, то другий інвертор отримає цей рівень на вході і, відповідно, видасть високий рівень сигналу на виході, і навпаки. Додатково до схеми додано два транзистори. Включаючи ці транзистори на вході та виході обох інверторів виходить логічний високий сигнал. Кожен період тактуючого сигналу, при відключенні транзисторів, обидва інвертори намагаються прийняти протилежне положення, тобто один із двох стійких станів, генеруючи при цьому один випадковий біт.

Дана розробка дозволила позбутися незручностей аналогових компонентів попереднього варіанта генератора випадкових чисел, значно зменшити енергоспоживання та збільшити швидкість генерації більш ніж у 30 тисяч разів [8].

На рис 1.6 зображена схема з двома невизначеними станами [16].

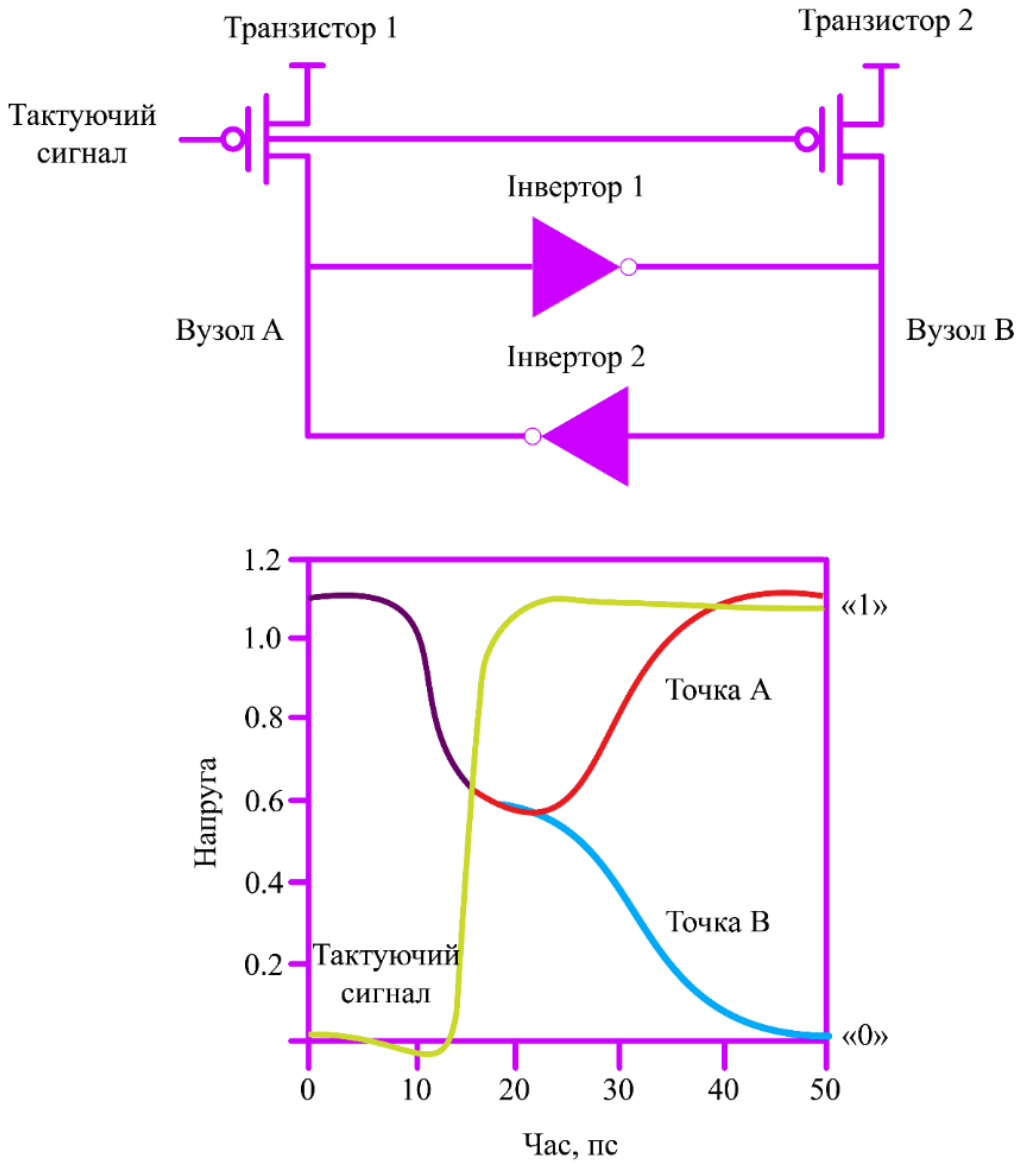

Рисунок 1.6 – Схема та діаграма сучасного генератора Intel

Транзистор 1 і транзистор 2 увімкнені, зв’язана пара інверторів, яка змушує вузол А і вузол В перевести в однаковий стан. Коли тактовий пульс підвищується (жовтий, праворуч), ці транзистори вимкнено. Спочатку вихід обох інверторів потрапляє в невизначений стан, але випадковий тепловий шум всередині інверторів незабаром штовхає один вузол у стан логічної 1, а інший переходить до логічного 0.

1.3 Вибір джерела шуму для реалізації апаратного генератора випадкових чисел

Автором було проведено порівняльний аналіз характеристик різних джерел шуму для подальшого вибору. Результати порівняння зведено в табл. 1.1.

Таблиця 1.1 – Характеристики джерел шуму

Характеристика / Назва джерела шуму	Відносна вартість реалізації приладу	Відносна складність реалізації	Використання у промислових зразках	Відносна швидкість генерації випадкових чисел
Дробовий шум	Мала	Мала	+	Мала
Тепловий шум (шум Джонсона)	Мала	Мала	+	Мала
Фазовий квантовий шум у лазерному промені	Висока	Висока	+	Висока
Лавинний шум	Мала	Мала	+	Мала
Фазове тремтіння в кільцевих генераторах	Середня	Мала	+	Мала
Спонтанне параметричне розсіювання	Висока	Висока	+	Середня
Радіоактивний розпад	Висока	Висока	-	Середня
Атмосферний шум	Середня	Висока	-	Середня
Матриця фотокамери	Середня	Середня	+	Дуже висока
Цифрова схема з невизначеним станом	Мала	Мала	+	Висока

З табл. 1.1 видно, що достатньо багато методів можна реалізувати доволі дешево. Це такі джерела шуму як: дробовий шум, тепловий шум, лавинний

шум та використання цифрової схеми з невизначеним станом. Серед найдорожчих вважається радіоактивний розпад, спонтанне параметричне розсіювання та фазовий квантовий шум у лазерному промені. Стосовно реалізації, то складніше реалізувати фазовий квантовий шум у лазерному промені, спонтанне параметричне розсіювання, радіоактивний розпад та атмосферний шум. Говорячи про відносну швидкість генерації випадкових чисел, то тут кращим є фазовий квантовий шум у лазерному промені та з використанням матриці фотокамери.

Серед усіх джерел шуму можна обрати дробовий або лавинний шум, але для даної роботи було обрано саме лавинний шум. Згідно отриманих значень цей метод не дуже дорогий та нескладний у реалізації. Хоча швидкість генерації не дуже висока, цей метод і досі можна використовувати у промислових зразках. Лавинний шум можна реалізувати за допомогою шумового стабілітрону з використанням мікроконтролера.

1.4 Висновки за розділом

Наведено основні поняття в сфері генерації випадкових чисел. Розглянуто 10 основних джерел шумів, а саме: дробовий шум, тепловий шум, фазовий квантовий шум у лазерному промені, лавинний шум, спонтанне параметричне розсіювання, радіоактивний розпад, атмосферний шум, шум матриці фотокамери та шум у цифровій схемі з невизначеним станом. Також проведений аналіз характеристик джерел шуму і для подальшої роботи обрано лавинний шум.

2 СКЛАД, ФУНКЦІЇ ТА РЕЖИМИ РОБОТИ КОМПЛЕКСУ

2.1 Структура комплексу

Дана робота представляє собою комплексну роботу яка включає два генератори випадкових чисел на основі мікроконтролера та мобільних пристроїв. Узагальнена структура даного комплексу зображена на рис. 2.1.

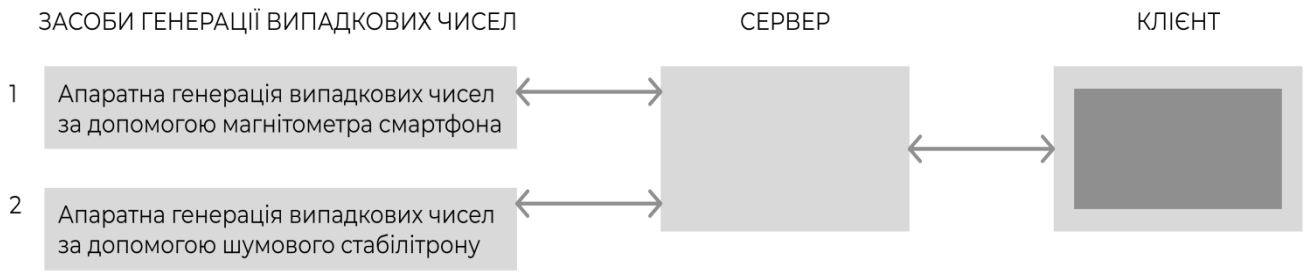

Рисунок 2.1 – Узагальнена структура комплексу

Сервер буде надавати можливість зручного вибору за командою від клієнтів типу генератору, який буде використовуватися для генерації випадкових чисел. Більш детальна структура зображена на рис. 2.2.

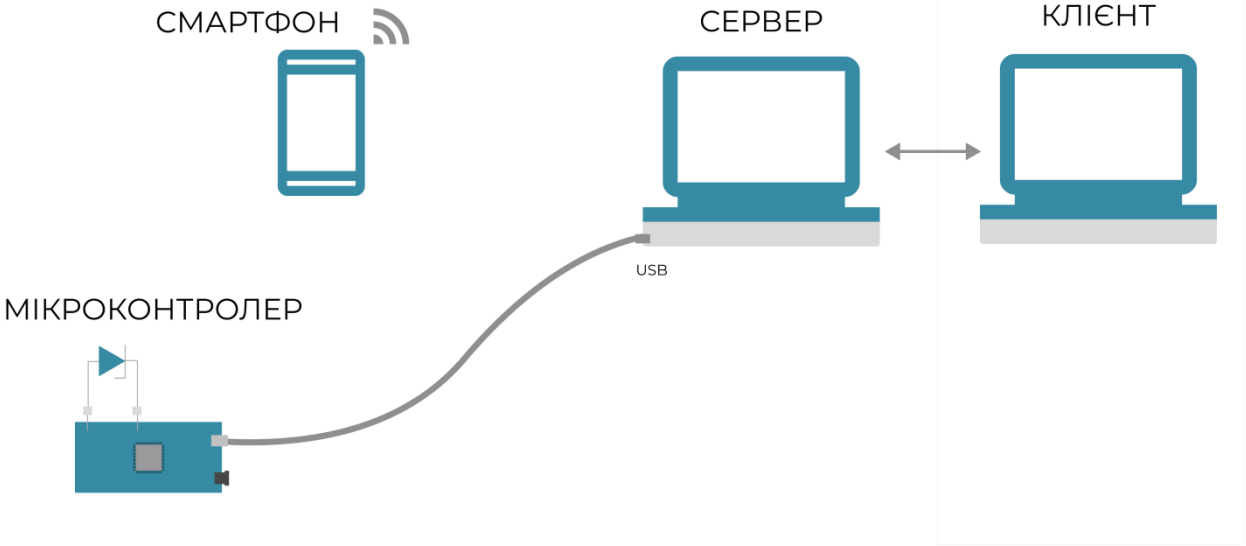

Рисунок 2.2 – Структура комплексу

Комплекс включає в себе мікроконтролер який отримує шум зі стабілітрону та смартфону який отримує шум з магнітометру. Також до комплексу включено сервер, який може використовуватися клієнтами як API для генерації випадкових чисел. Зв’язки між елементами комплексу зображено на рис. 1.3.

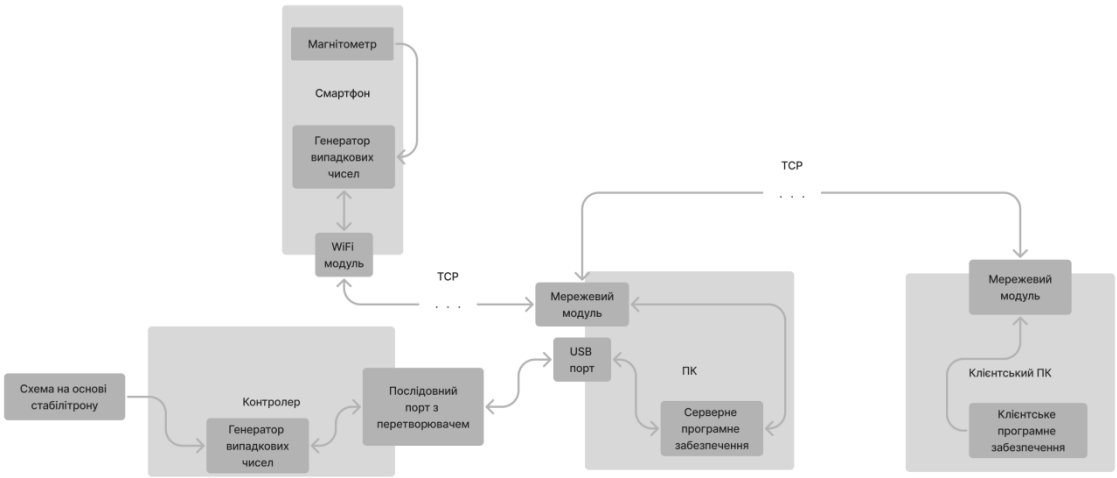

Рисунок 1.3 – Зв’язки між елементами комплексу

В якості протоколу для обміну між клієнтом та сервером обрано протокол TCP. Протокол TCP - це протокол, який може з’єднувати різні мережеві пристрої. Завдяки цьому протоколу програми у мережі можуть тримати зв'язок. Даний протокол має такі переваги [17]:

а) повторна передача даних (між транспортуванням сегменти можуть не досягнути місця призначення, тому одержувач надсилає підтвердження відправнику, щоб відправник міг повторно передати сегмент назад);

б) контроль перевантаження (TCP використовує окрему політику керування перевантаженнями - приймач надсилає сигнали відправнику, щоб уповільнити процес або затримати передачу);

в) унікальна ідентифікація (кожному комп’ютеру у протоколі TCP в мережі присвоєно унікальну IP-адресу);

г) виявлення помилок (пошкоджені та відсутні сегменти легко виявляться в TCP).

Даний сервер представляє собою TCP-сервер, який інтерпретує команди клієнта та генерує необхідну кількість випадкових чисел. Для генерації TCP-сервер використовує контролер та смартфон як генератори випадкових чисел.

Для реалізації запропоновано декілька режимів генерації випадкових чисел:

а) випадкові числа генерувати за допомогою апаратних;

б) псевдовипадкові числа генерувати апаратно згенерованим зерном;

в) псевдовипадкові числа генерувати за допомогою алгоритму.

У якості алгоритму генерації псевдовипадкового числа обирано алгоритм Blum-Blum-Shub (BBS). Даний алгоритм видає дійсно якісну послідовність псевдовипадкових чисел з великим періодом, має достатню криптостійкість та легкість реалізації .

Кроки реалізації алгоритму [18]:

а) згенерувати 2 досить великі секретні випадкові (і різні) прості числа

p,q ≡3(mod4) і порахувати N = p * q;

б) вибрати випадкове початкове ціле число s ($1 \leq s \leq N - 1$) отже

НОД(s, N)= 1. Обчислити $u_0 = s^2 \bmod N$;

в) для i = $\overline{1, m}$:

1) обчислити $u_i = u_i^2 \bmod N$;

2) обчислити x_i як молодший біт двійкового представлення числа u_i.

г) у результаті попереднього кроку формується вихідна псевдовипадкова послідовність $x_1, x_2, \ldots x_m$.

Для взаємодії між сервером та контролером використовується послідовний інтерфейс. В цьому комплексі сервер надсилає кількість необхідних чисел до контролера, який відповідає згенерованими числами. Аналогічна взаємодія відбувається між сервером та смартфоном, єдина різниця це те, що в цьому випадку дані передаються по TCP через WiFi.

2.2 Організація обміну між елементами комплексу

У роботі реалізовано взаємодію між мікроконтролером та сервером. Коли сервер отримує команду від клієнта він з цієї команди бере кількість чисел, яку треба згенерувати. Це число потім відправляється по порту на мікроконтролер. Формат запиту це число з символом кінця рядка (\n). У відповідь сервер отримує послідовно згенеровані числа, які розділені символом кінця рядка.

В даній роботі розроблено систему команд взаємодії між клієнтом та сервером. Ці дані наведені у табл.2.1.

Таблиця 2.1 – Параметри команди обміну між клієнтом та сервером

Команда	Аргумент 1 – метод генерації	Аргумент 2 – необхідна кількість випадкових чисел	Опціональний аргумент 3 – ідентифікатор апаратного генератору		Можливі відповіді	Інтерпретація
GEN	TR - всі числа згенеровані апаратно	[1; 255]	SP - смартфон	MK - мікроконтролер	GEN[згенеровані числа]	Згенеровані числа
	MR – апаратне зерно з псевдовипадковою послідовністю				FMT	Неправильний формат команди
	PR - псевдовипадкова послідовність				CON	Необхідний апаратний генератор не підключений

Виходячи з даних у таблиці можна зробити такі запити до TCP серверу:

а) GENTRSP10 – 10 випадкових чисел апаратно згенерованих за допомогою смартфона;

б) GENMRMK20 – 20 псевдовипадкових чисел з апаратно згенерованим зерном за допомогою мікроконтролера.

в) GENPR30 – 30 псевдовипадкових чисел.

Можливі відповіді сервера наведені у табл.. 2.2.

Таблиця 2.2 – Можливі відповіді сервера

Можливі відповіді	Опис
GEN[згенеровані числа у hex]	Числа успішно згенеровані
FMT	Невірний формат запиту
CON	Вказаний апаратний генератор непідключений

Виходячи з даних у таблиці можна отримати такі відподі від сервера :

а) GEN 187222 21822 370 – згенеровані випадкові числа;

б) FMT – це означає, що клієнт відправил невірну команду;

в) CON – це означає, що даний апаратний генератор не підключений

2.3 Принцип оцифрування шумів

В даному комплесі пропонується отримувати випадкові числа за допомогою використання порогового значення. Отримані випадкові величини у генераторі коливаються біля одного значення. Пропонується зняти приблизно 1000 точок та обрати максимальне та мінімальне значення, яке приймає ця випадкова величина. Після чого максимальне і мінімальне сумувати та визначити порогове значення. Завдяки цьому, можна визначити у який момент буде 0 або Після отриманних випадкових нулів та одиниць формуємо двійкову послідовність, а потім розбиваємо на випадкові 32-бітні числа. Які потім відправляємо на сервер клієнту.

На рис. 2.1 представлений графік для визначення порогового значення.

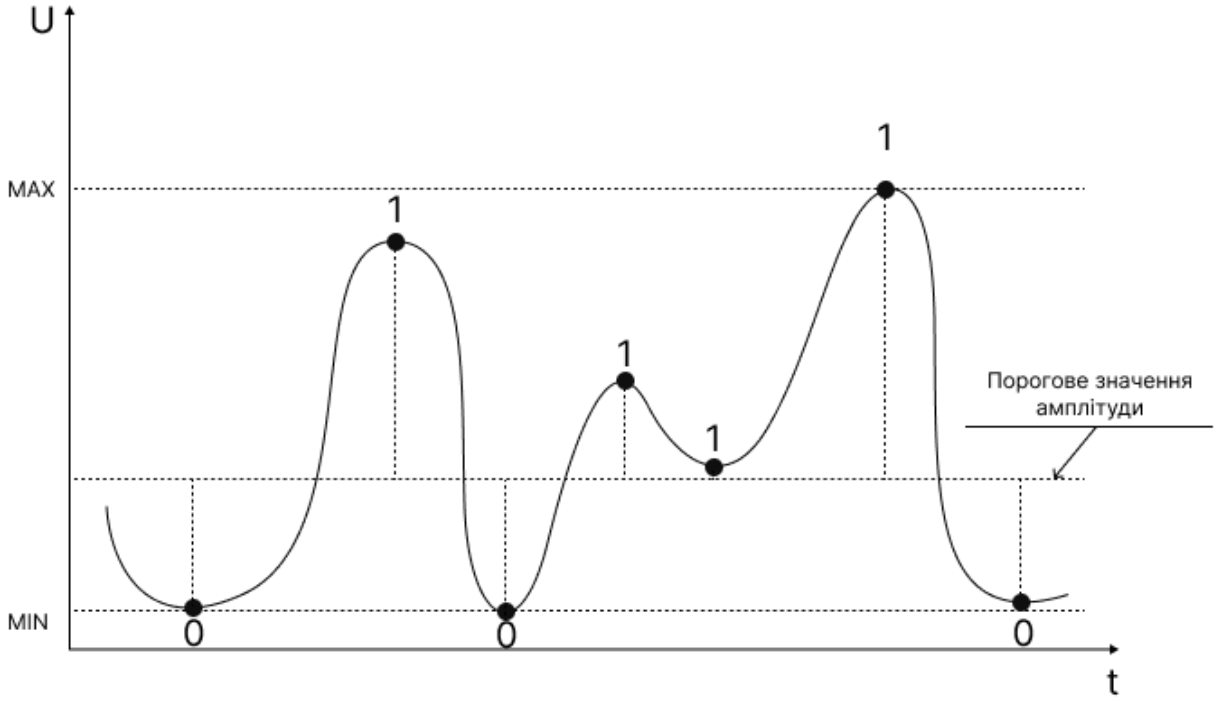

Рисунок 2.1 – Графік для визначення порогового значення

2.4 Висновки за розділом

Наведено структуру комплексу та показано взаємодію між його елементами. Розроблено систему команд для обміну між клієнтом та сервером та можливі відповіді сервера на запити. Розглянуто принцип оцифрування значень шумів.

3 РОЗРОБКА АПАРАТНОЇ ЧАСТИНИ КОМПЛЕКСУ

3.1 Розробка схеми пристрою

Для створення апаратного генератора випадкових чисел обран: джерело шуму, мікроконтролер, USB-перетворювач для зв'язку мікроконтролера та комп’ютеру. Дана схема приведена на рис. 3.1

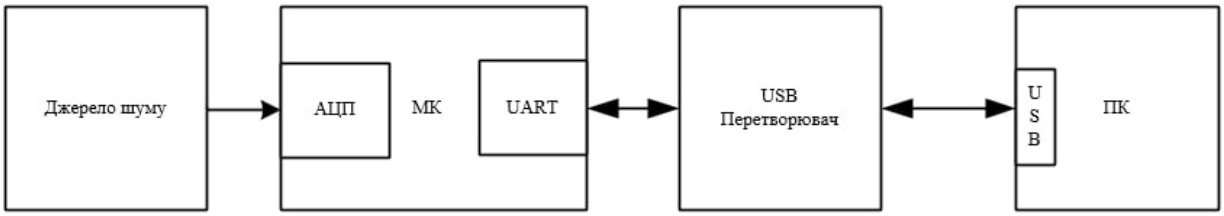

Рисунок 3.1 – Загальна структура комплексу на базі мікроконтролеру

Обрано для даного комплексу плату Arduino Mega 2560 на базі мікроконтролера ATmega2560. Плата має 54 цифрових вхідних/вихідних контакти, 16 аналогових входів, 4 UART (апаратні послідовні порти), кристалічний осцилятор 16 МГц, USB-з'єднання, роз’єм живлення, заголовок ICSP та кнопку скидання.

Дана плата Arduino Mega 2560 має такі переваги:

а) використовується для складання та моделювання прототипів електронних пристроїв на базі платформ Arduino стандартного розміру (Uno, Leonardo, Duemilanove, Diecimila, Extreme, NG, WeMos D1 R2 V2.1.0) та інших сумісних плат;

б) простий процес розробки схем;

в) простий процес налагодження схем;

г) є можливість швидко та надійно фіксувати компоненти;

д) наявність універсальний асинхронного приймача/передавача (UART);

е) наявність аналого-цифрового перетворювача;

ж) живлення макетної плати може здійснюватися як з Arduino контролера, і від зовнішніх джерел живлення (блоків живлення, батарей).

Плата Arduino Mega 2560 зображена на рис. 3.2 [19].

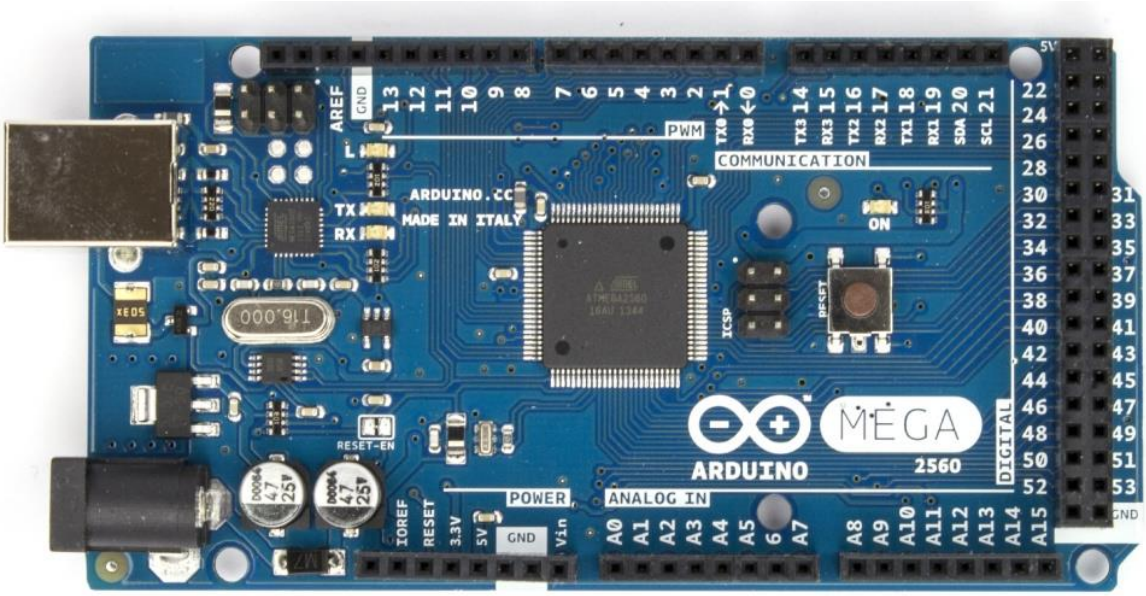

Рисунок 3.2 – Arduino Mega 2560

Дана плата має такі характеристики наведені у табл. 3.1 [20]:

Таблиця 3.1 Характеристики Arduino Mega 2560

Характеристика	Значення
Мікроконтролер	ATmega2560
Робоча напруга	5В
Напруга живлення (рекомендоване)	7-12В
Напруга живлення (гранична)	6-20В
Цифрові входи/виходи	54
Аналогові входи	16
Максимальний струм одного виводу	40 мА
Максимальний вихідний струм виводу 3.3V	50 мА
Flash-пам'ять	256 КБ
SRAM	8 КБ
EEPROM	4 КБ
Тактова частота	16 МГц

У якості джерела шуму обрано стабілітрон КС133Г. Даний стабілітрон має такі технічні характеристики[21]:

а) номінальна напруга стабілізації: 3,3 В при Іст 5 мА;

б) розкид напруги стабілізації: 2,95...3,65 В;

в) диференціальний опір стабілітрону: 150 Ом при Іст 5 мА;

г) мінімально допустимий струм стабілізації: 1 мА;

д) максимально допустимий струм стабілізації: 37,5 мА;

е) максимально-допустима потужність, що розсіюється, на стабілітроні: 0,125 Вт;

ж) робочий інтервал температури навколишнього середовища: -60...+125 °С.

Для даного комплексу обрано плату з робочою напругою 5В. Виходячи з параметрів, перерахованих вище, стабілітрон КС133Г можна використовувати на платі Arduino Mega 2560 без використання додаткових елементів, тому що напруга стабілізації на стабілітроні до 3,65 В.

У даному проекті обрано резистор номіналом 1кОм. За його допомогою можна обмежити струм.

Для підключення до плати стабілітрона та резистора використовували такі порти живлення 5В, заземлення GND та аналоговий порт А1.

Схема підключення джерела шуму до Arduino Mega 2560 зображена на рис.2.4.

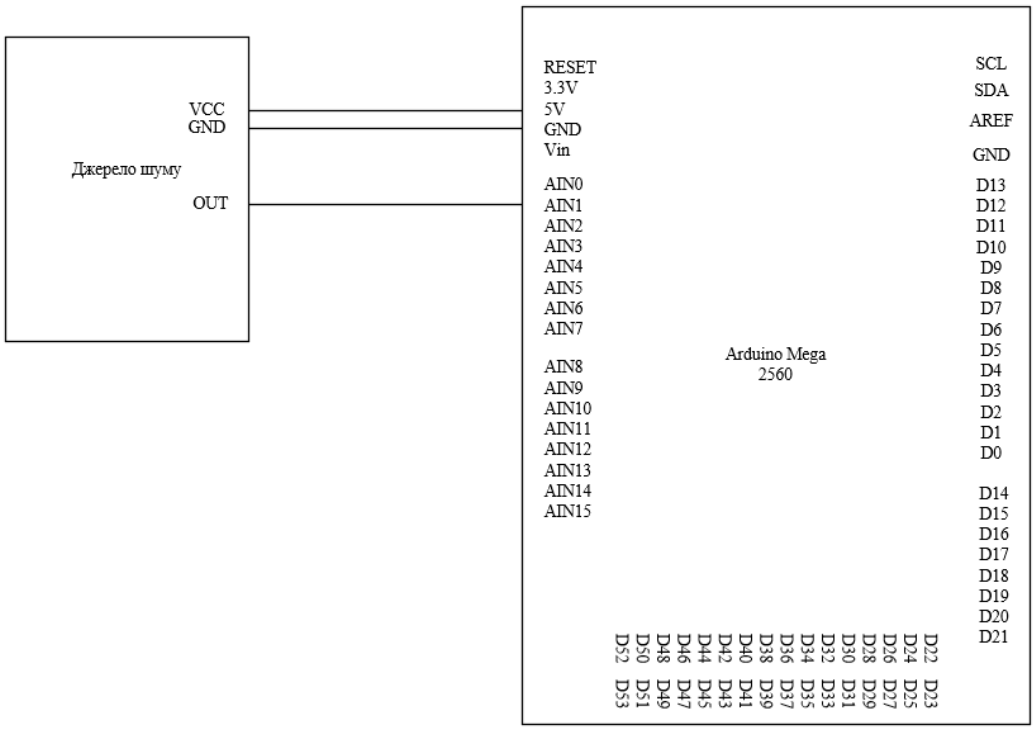

Рисунок 3.3 – Схема підключення джерела шуму до Arduino Mega 2560

На рис. 3.4 зображена принципіальна схема джерела шуму. Даний напівпровідниковий прилад виробляє електричні коливання, які після посилення створюють шум. Схема включає в себе стабілітрон КС133Г, який взятий для генерації шуму. Резистор R1 обрано достатньо великим, щоб забезпечити режим роботи стабілітрону. Два посилювача напруги (DD1.1 та DD1.2). Для переводу у лінійний режим вони огорнуті зворотнім зв'язком за постійним струмом з виходу на вхід через резистори R2 та R4.

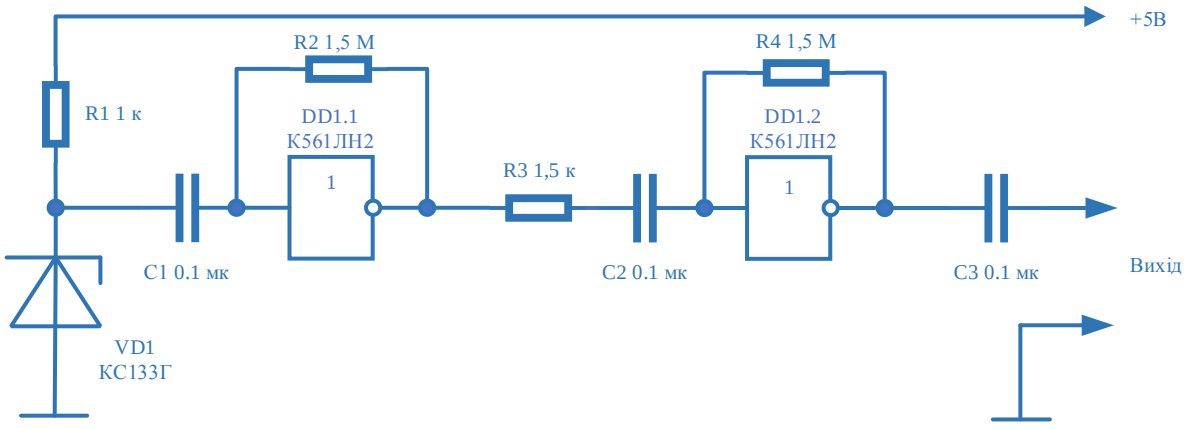

Рисунок 3.4 – Принципова схема джерела шуму

Шумовий сигнал на стабілітроні КС133Г проходить через конденсатор С1 та надходить на вхід посилювача DD1.1 К561ЛН2, а з його виходу проходить резистор R3 з конденсатором С2 та знову потрапляє на вхід підсилювача, але вже на DD1.2 К561ЛН2. Посилений сигнал проходить через конденсатор С3 і вже прямує на вихід пристрою.

У результаті роботи зібрано таку схему, що показана на рис. 3.5.

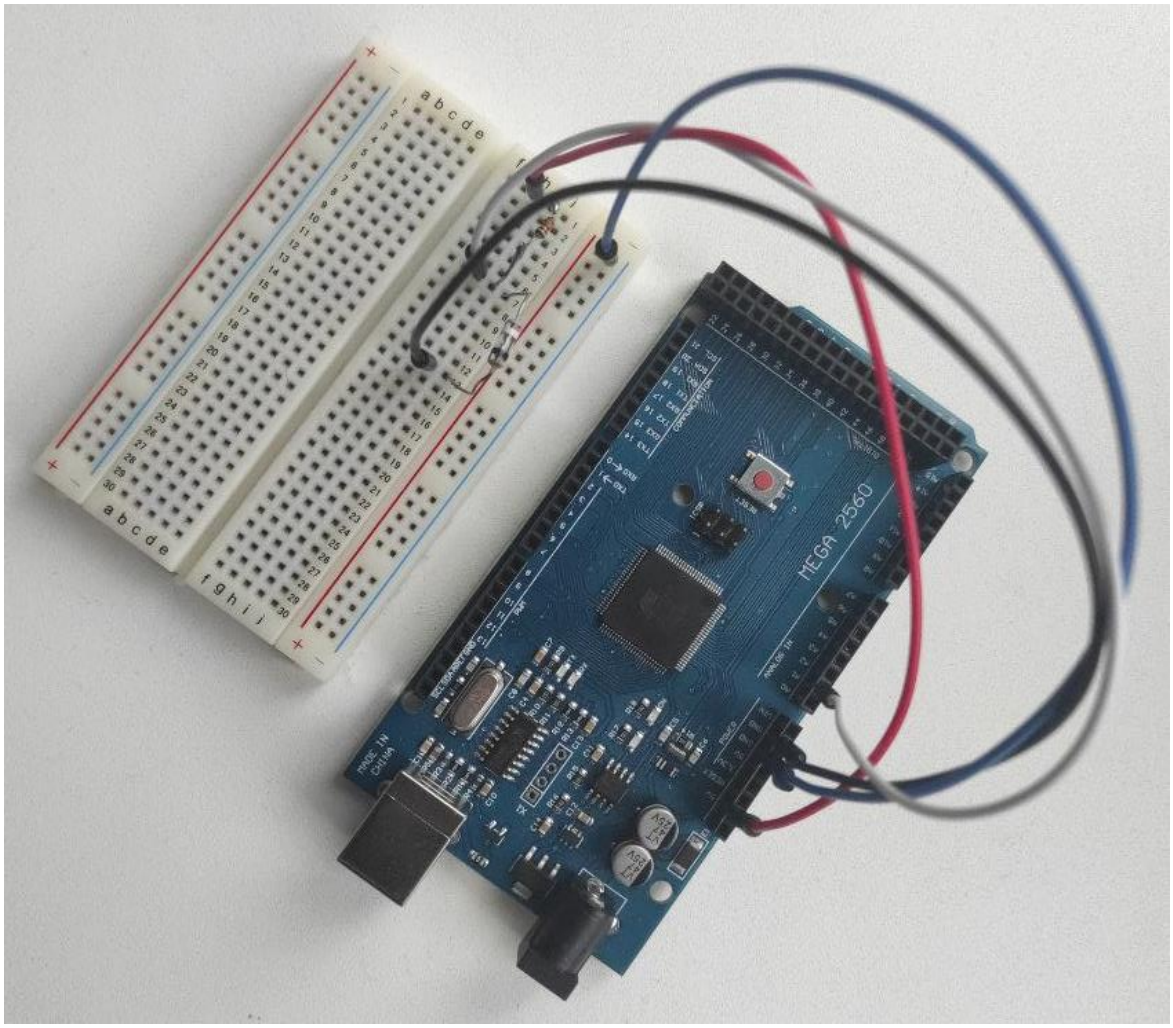

Рисунок 3.5 – Зовнішній вигляд макету пристрою

На даному рисунку видно, що стабілітрон КС133Г підключений послідовно з резистором 1кОм. Поміж резистором та стабілітроном встановлений дріт,

який підключений до аналогового порту A1 на Arduino Mega 2560 та зчитує дані напруги.

3.2 Розробка програмного забезпечення пристрою

Для реалізації комплексу генерації випадкових чисел обрано інтегроване середовище розробки Arduino IDE. Зовнішній вигляд середовища зображено на рис. 3.6. Програмне забезпечення Arduino IDE має сучасний та зручний інтерфейс, має легку навігацію, живий налагоджувач та автозаповнення. Ця програма має достатньо багато прикладів реалізації простих проектів, багата на бібліотеки, є можливість легко програмувати плату з мікроконтролером за допомогою кабелю [22]. Тому обрано саме Arduino IDE для програмування плати Arduino Mega 2560.

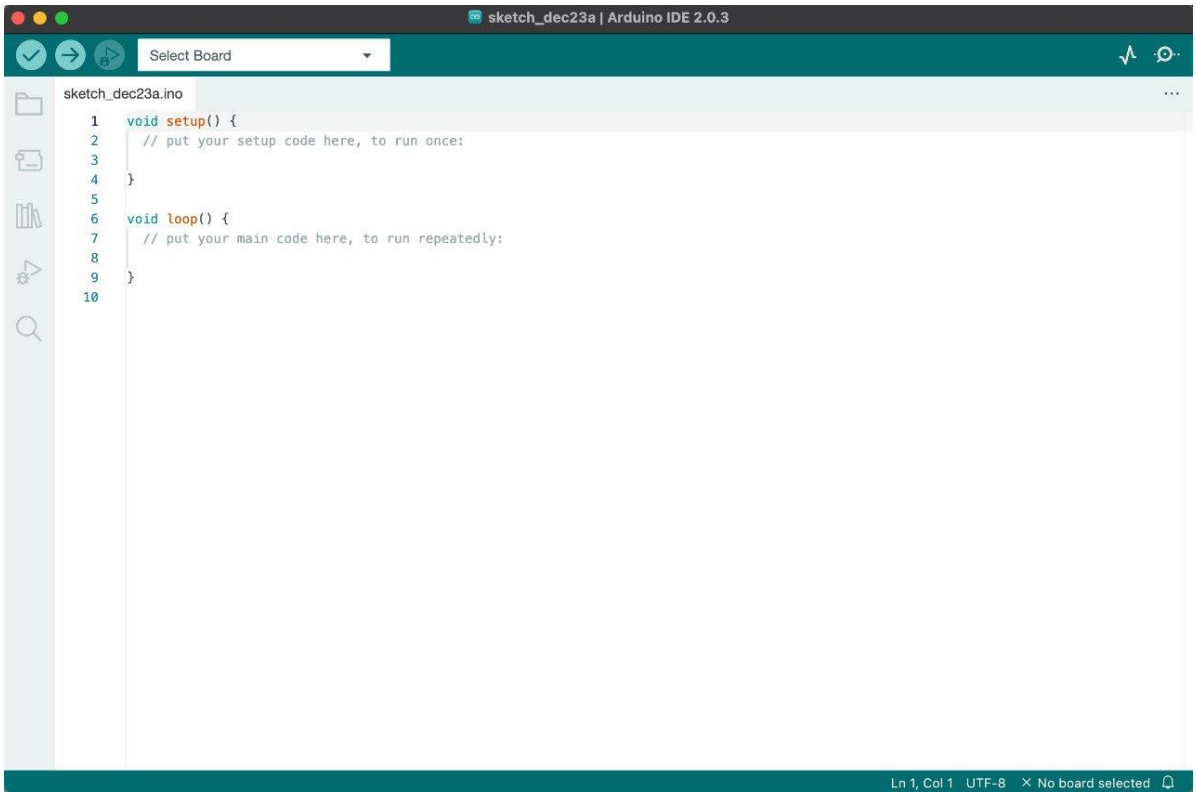

Рисунок 3.6 – Зовнішній вигляд середовища Arduino IDE

Була розроблена програма для генерації випадкових чисел у середовищі Arduino IDE. Програма складається з чотирьох основних функцій:

а) setup();

б) loop();

в) randomInt32();

г) serialEvent().

Функція setup використовується для ініціалізації змінних та визначення режимів роботи виходів.

Після виклику функції setup(), запускається функція loop() та викликає нескінченний цикл. Узагальнений алгоритм наведено на рис. 3.7.

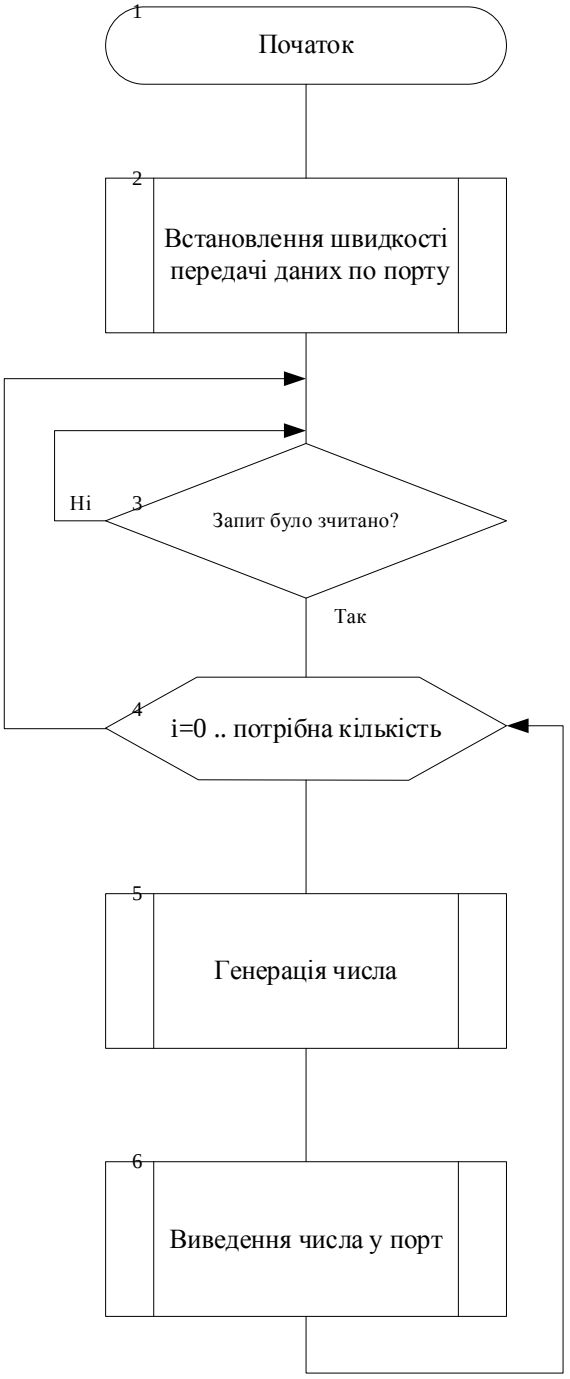

Рисунок 3.7 – Узагальнений алгоритм ініціалізації та основного циклу

Блок 1: Початок роботи функції.

Блок 2: Ініціалізація змінних.

Блок 3: Перевірка чи було зчитано кількість чисел для генерації.

Блок 4: Цикл FOR для генерації кожного числа.

Блок 5: Процедура генерації числа.

Блок 6: Вивід числа у порт.

Функція serialEvent автоматично викликається в кінці, коли в буфері є послідовні дані. Блок-схема цієї функції наведено на рис. 3.8.

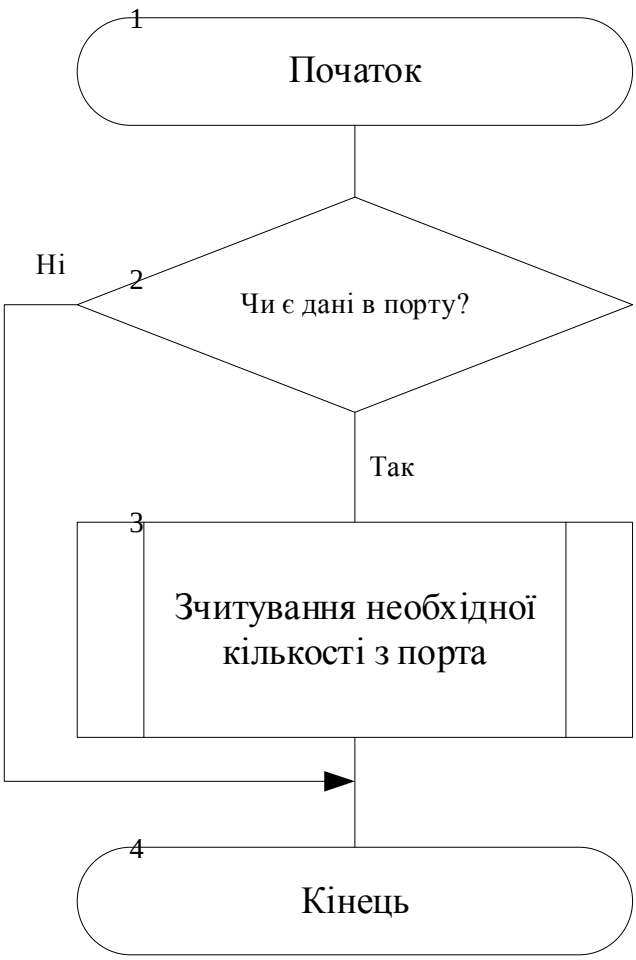

Рисунок 3.8 – Узагальнений алгоритм обробки події від послідовного порту

Блок 1: Початок роботи функції.

Блок 2: Перевірка чи є дані в порту.

Блок 3: Процедура зчитування необхідної кількості з порта.

Блок 4: Кінець роботи функції.

Розроблено програмне забезпечення пристрою у середовищі Arduino IDE, програма якої наведена у додатку Б.

3.3 Висновки за розділом

Для створення пристрою генерації випадкових чисел обрано Arduino Mega 2560. У якості джерела шуму обрано стабілітрон КС133Г. Наведено схему підключення джерела шуму. Проведена розробка програмного забезпечення пристрою.

4 РОЗРОБКА ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СЕРВЕРНОЇ ТА КЛІЄНТСЬКОЇ ЧАСТИНИ

4.1 Вибір середовища та засобів розробки

Для реалізації даного проекту обрано мову програмування Golang. Дана мова має такі переваги [23,24]:

а) швидкість (Golang здатний компілювати безпосередньо в машинний код без використання інтерпретатора);

б) легкий у навчанні (простий для розробників програмного забезпечення, особливо для тих, хто вже добре розуміє C і Java);

в) можливість маштабувати (здатність підтримувати паралельне програмування);

г) велика кількість бібліотек;

д) велика екосистема партнерів, спільнот та інструментів.

У якості середовища розробки було обрано Goland IDE. Вікно середовища Goland IDE представлено на рис. 4.1.

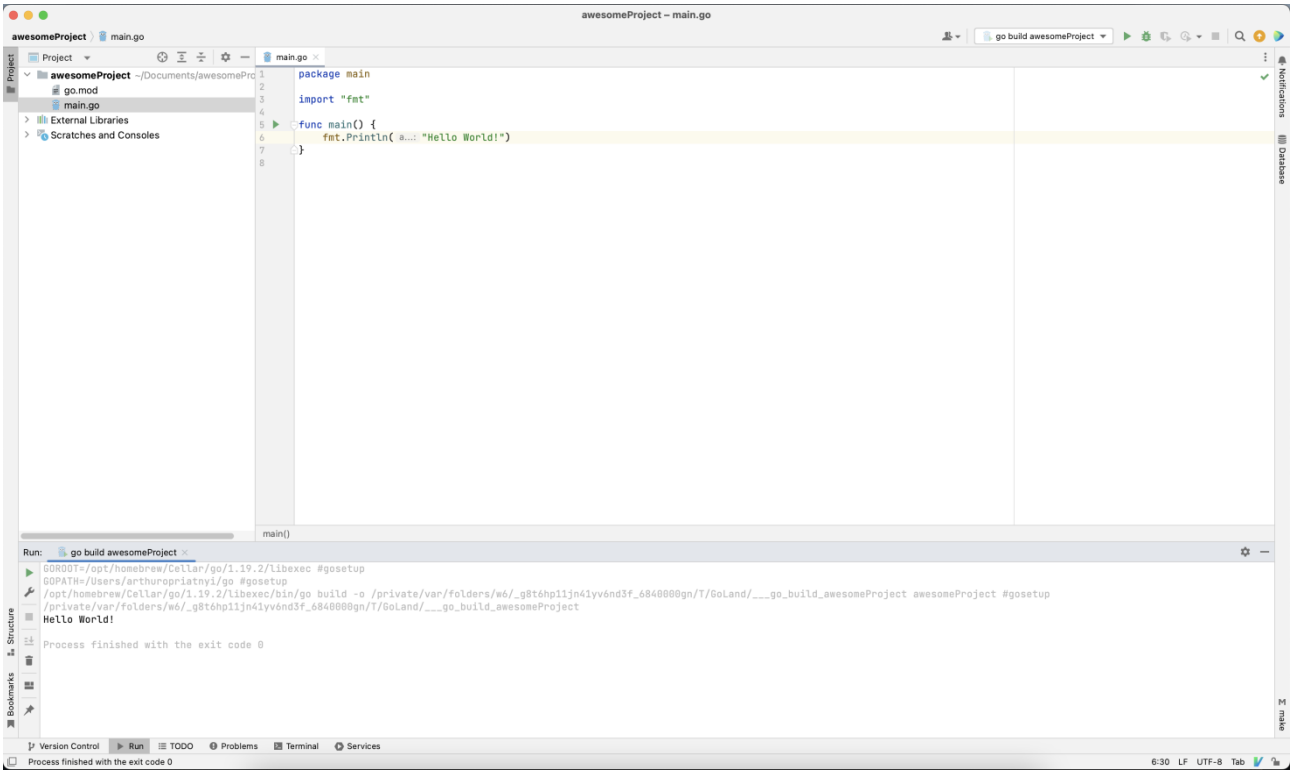

Рисунок 4.1 – Середовище розробки Goland IDE для мови програмування Golang

Середовище Goland IDE має такі переваги [25]:

а) GoLand робить розробку простішою та ефективнішою;

б) є можливість миттєво переходити до будь-якого елемента в коді;

в) налагодження та запуск безпосередньо в IDE без установки плагінів або додаткової конфігурації;

г) вбудована підтримка Git, GitHub і Mercurial;

д) широкий вибір інструментів;

е) включає понад 1000 плагінів, за допомогою яких можна адаптувати IDE до своїх потреб.

В даній роботі у середовищі Goland IDE використовується бібліотека Serial [26]. Ця бібліотека була обрана для роботи з послідовними портами.

4.2 Розробка програми сервера та клієнта

Для даного комплексу розроблено програму сервер у середовищі Goland IDE. Код програми наведено у додатках В, Г, Д, Е, Ж, И, К, Л. Нижче приведена блок-схема цієї програми на рис.4.2:

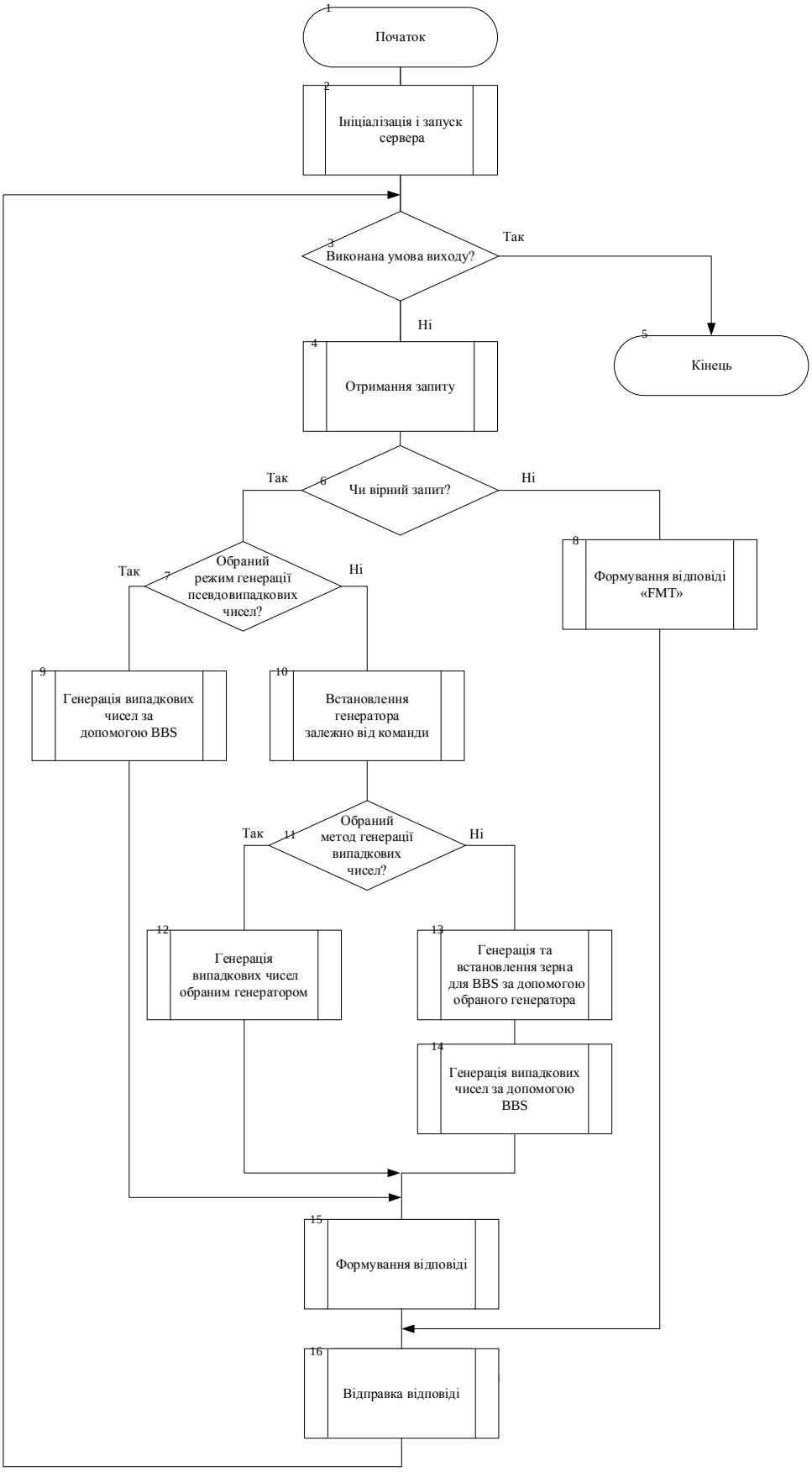

Рисунок 4.2 – Узагальнений алгоритм роботи серверу для обробки запиту від одного клієнта

Блок 1: Отримання запиту сервером.

Блок 2: Перевірка правильності запиту.

Блок 3: Перевірка чи було обрано режим генерації псевдовипадкових чисел.

Блок 4: Формування відповіді для неправильно сформованої команди.

Блок 5: Генерація випадкових чисел за допомогою (Блюм–Блюма–Шуба (BBS**).**

Блок 6: Встановлення генератора залежно від команди.

Блок 7: Перевірка чи було обрано режим генерації випадкових чисел.

Блок 8: Генерація випадкових чисел обраним генератором.

Блок 9: Генерація та встановлення зерна для BBS за допомогою обраного генератора.

Блок 10: Генерація випадкових чисел за допомогою BBS.

Блок 11: Формування відповіді.

Блок 12: Відправка відповіді.

Для даного комплексу було розроблений приклад клієнтського застосунку. Ця програма дуже проста у реалізації, тому не потребує демонстрації алгоритму. Користувач вводить команду у клієнтській програмі, далі клієнт встановлює TCP-з'єднання з сервером і відправляє йому зчитану команду. Сервер виконує цю команду та відправляє її клієнту. Клієнт виводить результат у консоль, закриває TCP-з'єднання та завершує роботу.

4.3 Налагодження роботи програми

Для налагодження розробленого програмного забезпечення можна скористатися наступною інструкцією з його використання.

Для запуску сервера використовується команда go run main.go.

Для зупинки використовується комбінація клавіш Ctrl + C.

Для запуску клієнта використовується команда ADDRESS=address CMD=cmd go run main.go, де

а) address – IP-адрес сервера з портом;

б) cmd – команда, яка буде відправлена на сервер.

На рис. 4.2 – 4.7 наведено різні результати під час налагодження програми.

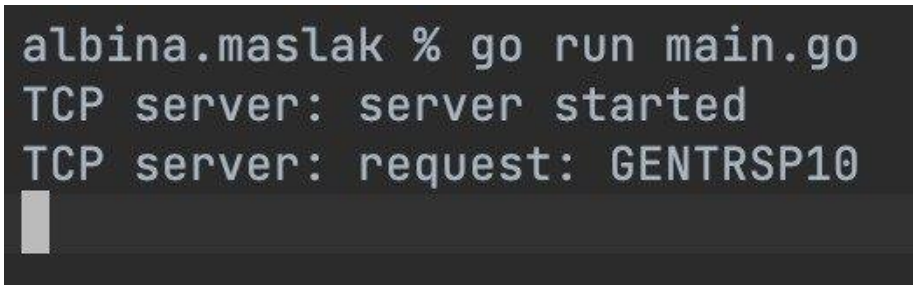

Рисунок 4.2 – Запуск сервера та обробка команди, яка до неї прийшла

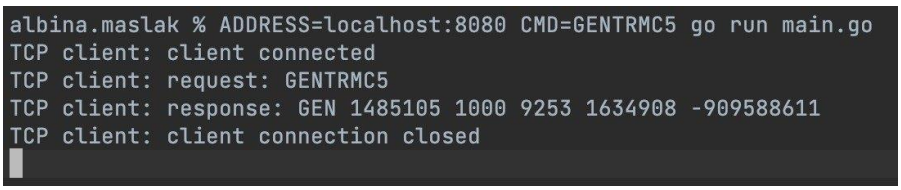

Рисунок 4.3 – Отримання клієнтом 5 випадкових чисел за допомогою мікроконтролера

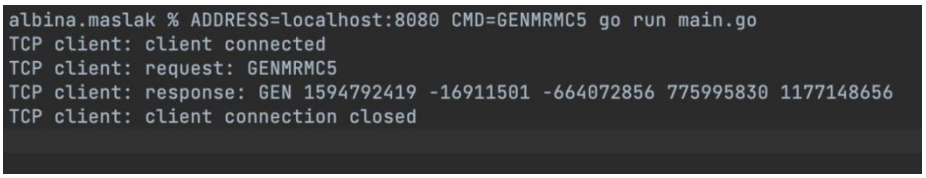

Рисунок 4.4 – Отримання клієнтом 5 псевдовипадкових чисел використовуючи зерно згенероване за допомогою мікроконтролера

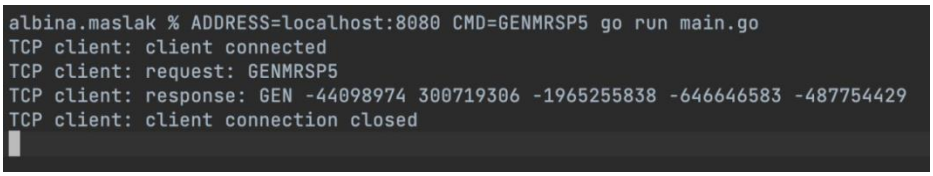

Рисунок 4.5 – Отримання клієнтом 5 псевдовипадкових чисел використовуючи зерно згенерований за допомогою смартфона

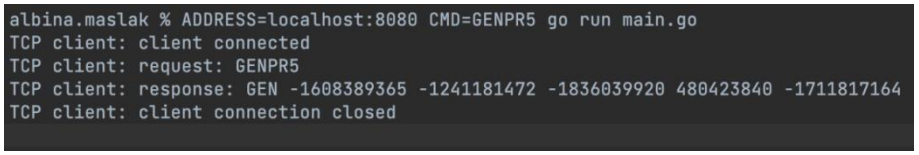

Рисунок 4.6 – Отримання клієнтом 5 псевдовипадкових чисел

```
albina.maslak % ADDRESS=localhost:8080 CMD=GENPR go run main.go
TCP client: client connected
TCP client: request: GENPR
TCP client: response: FMT
TCP client: client connection closed
```

Рисунок 4.7 – Отримання клієнтом інформації, що команда з неправильним форматом

4.4 Висновки за розділом

Обрано середовище розробки Goland IDE та мову програмування Golang. Розроблені основні алгоритми, також розроблено програмне забезпечення серверної частини. Для налагодження сервера використано приклад клієнтської частини.

5 ДОСЛІДЖЕННЯ ЯКОСТІ ОТРИМУВАНИХ ВИПАДКОВИХ ЧИСЕЛ

5.1 Перевірка випадкових чисел статистичним тестом випадковості

Використовуючи генератор на базі мікроконтролера проведено тестування отриманих випадкових чисел. Для дослідження статистичним тестом було згенеровано 2000 випадкових чисел. Для зручності відображення згенерованих чисел у клієнта використовується десяткова система, однак для подальшого тестування необхідно відображати числа, як двійкову послідовність. Тому вирішено зберігати файл з числами у форматі двійкової послідовності в окремий файл. В кожному файлі один рядок це одне 32-бітне число, як у десятковому вигляді, так і у двійковому. Приклад чисел у десятковому виді наведено на рис.5.1. Згенеровані числа у двійковому форматі приведені у додатку М.

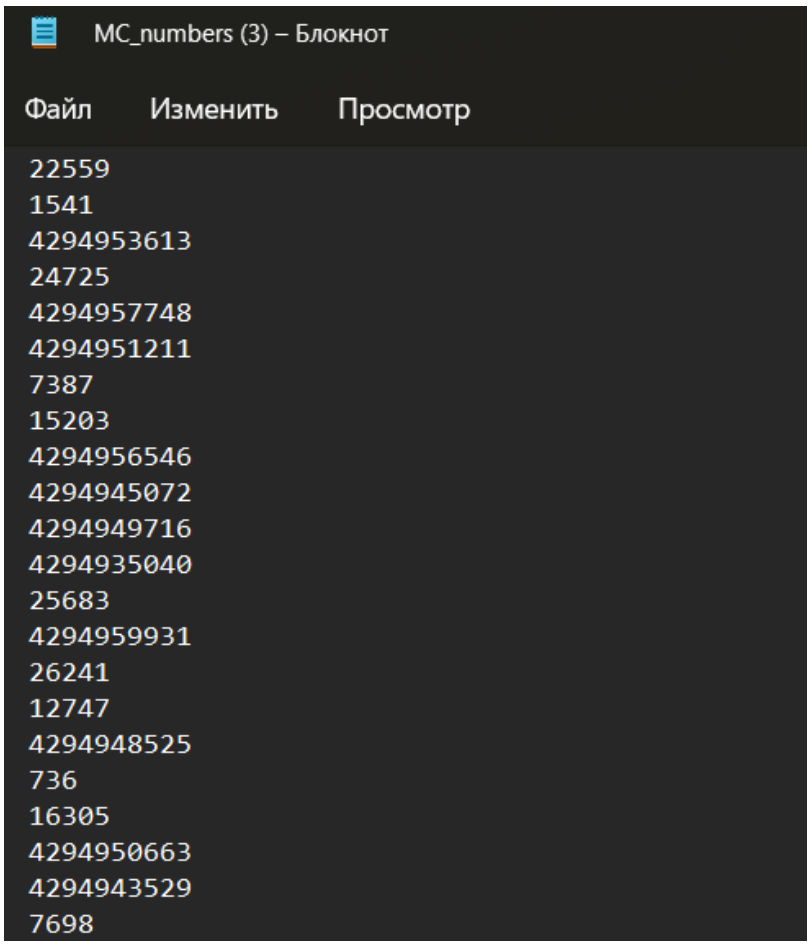

Рисунок 5.1 – Приклад файлу зі згенерованими числами у десятковому форматі

Приклад чисел згенерованих з використанням розроблено пристрою наведено на рис. 5.2.

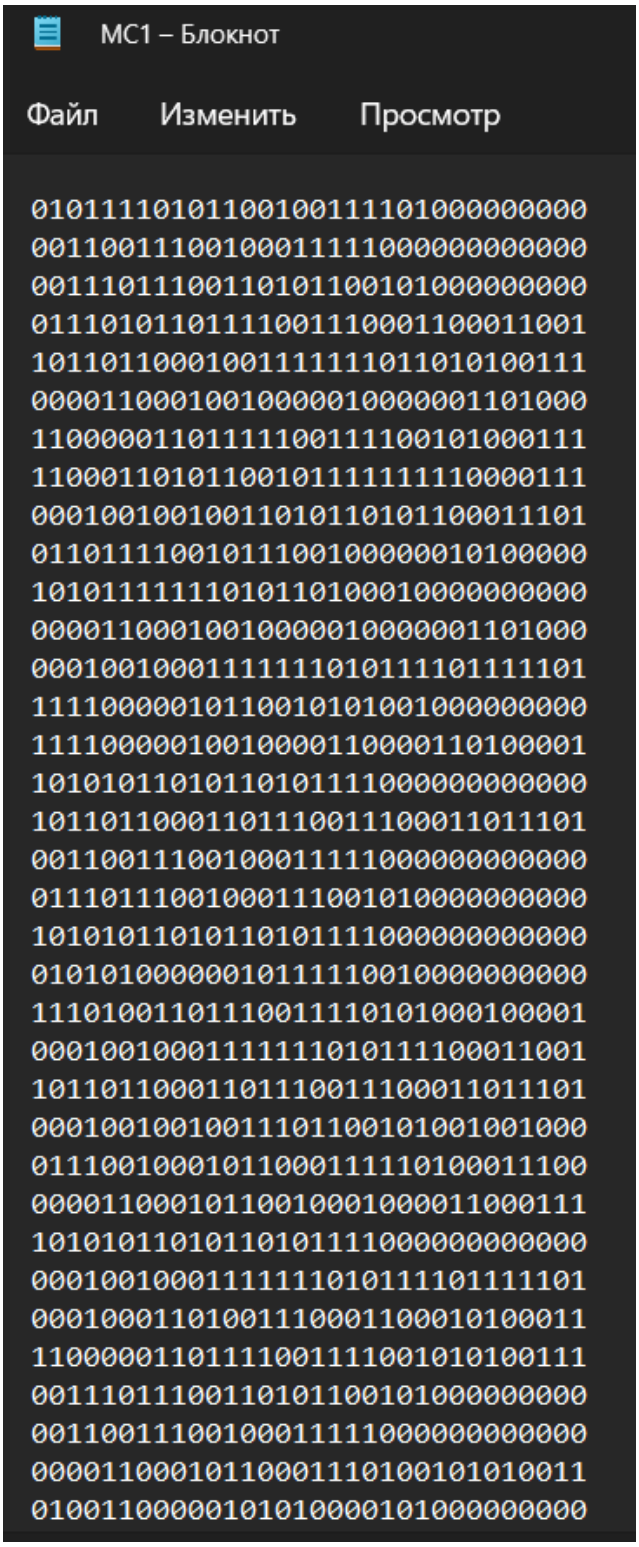

Рисунок 5.2 – Приклад вмісту файлу з випадковими числами у двійковій формі

Для перевірки на випадковість отриманих за допомогою розробленого пристрою чисел в даній роботі обрано статистичний пакет тестів NIST [27]. Даний пакет складається з 15 тестів, які розроблені для перевірки випадковості двійкових послідовностей, що генеруються апаратними або програмними криптографічними генераторами випадкових або псевдовипадкових чисел. Також тести орієнтовані на різні типи невипадковостей, які можуть з'явитися в згенерованій послідовності.

Статистичний пакет NIST складається з таких тестів:

а) частотний побітовий тест;

б) частотний блоковий тест;

в) тест на послідовність однакових бітів;

г) тест на найдовшу послідовність одиниць у блоці;

д) тест рангів бінарних матриць;

е) спектральний тест;

є) тест на збіг шаблонів, що не перекриваються;

ж) тест на збіг шаблонів, що перекриваються;

з) універсальний статистичний тест Маурера;

и) тест на лінійну складність;

і) тест на періодичність;

ї) тест приблизної ентропії;

й) тест кумулятивних сум;

к) тест на довільні відхилення;

л) інший тест на довільні відхилення.

Частотний тест — це тест, який визначає кількість одиниць і нулів у послідовності та перевіряє кількість одиниць і нулів приблизно така ж, як у випадковій послідовності [28]. Іншими словами, цей тест визначає, чи наближається до 1/2. Тобто кількість одиниць і нулів у досліджуваній послідовності має бути приблизно однаковою.

Суть тесту на послідовність однакових бітів полягає у визначенні частки одиниць в межах блоку з довжиною m бітів. Мета полягає в тому, щоб

з'ясувати, чи швидкість повторення одиниць в блоці довжини бітів m приблизно дорівнює m/2, як це можна було б припустити в разі абсолютно випадкової послідовності. Значення ймовірності p, розраховане в ході тесту, має бути не менше 0, 01. В іншому випадку (p < 0,01) бінарна послідовність не є по-справжньому випадковою.

Суть тесту на послідовність однакових бітів полягає в обчисленні загальної кількості рядків у вихідній послідовності. Тут слово «ряд» означає безперервну підпослідовність ідентичних бітів. Послідовність довжиною k-бітів складається з k-однакових бітів, починаючи і закінчуючи бітом, що містить протилежне значення. Мета цього тесту полягає в тому, щоб зробити висновок, чи відповідає кількість рядків одиниць і нулів різної довжини кількості випадкових послідовностей.

Тест для перевірки найдовшої послідовності одиниць у блоці визначає найдовшу послідовність одиниць у блоці довжиною m біт. Мета полягає в тому, щоб з'ясувати, чи справді довжина такого ряду відповідає очікуваній довжині найдовшого ряду одиниць у випадку абсолютно випадкової послідовності. Якщо тест обчислює значення ймовірності p < 0,01, вихідна послідовність вважається невипадковою. В іншому випадку він вважається випадковим. Зауважте, що припущення, що 1s і 0s зустрічаються приблизно з однаковою частотою, дає точно той самий результат для цього тесту, якщо ми розглядаємо найдовшу послідовність 0s. Тому вимірювання можна виконувати лише в одиницях вимірювання [29].

Перевірка рангу двійкової матриці обчислює ранг підматриць, що не перетинаються, побудованих з вихідних двійкових послідовностей. Метою цього тесту є перевірка лінійних залежностей фіксованої довжини, які складають вихідну послідовність. Якщо значення ймовірності p, обчислене під час тесту, < 0,01, ми робимо висновок, що вхідна послідовність бітів не є випадковою. В іншому випадку він вважається абсолютно випадковим [29].

Суть спектрального тестування полягає в оцінці висот піків дискретного перетворення Фур'є вихідної послідовності. Мета полягає в виявленні

періодичних властивостей у вхідній послідовності, таких як тісно розташовані повторювані області. таким чином чітко вказуючи на відхід від випадкової природи досліджуваних послідовностей. Ідея полягає в тому, що кількість піків вище 95% амплітудного порогу має бути значно більше 5%. Якщо значення ймовірності p, обчислене під час тестування, < 0,01, то ця двійкова послідовність не є абсолютно випадковою. інакше це випадково.

Унікальний тест зіставлення шаблонів підраховує кількість визначених шаблонів, знайдених у вихідній послідовності. Мета полягає в тому, щоб визначити генератори випадкових або псевдовипадкових чисел, які генерують часто задані нерегулярні шаблони. Подібно до зіставлення шаблонів перекриття для шаблонів, що перекриваються, ми використовуємо вікно довжиною м біт для пошуку конкретного шаблону довжиною м біт. Якщо шаблон не знайдено, вікно зсувається на 1 біт. Якщо шаблон знайдено, вікно переходить до наступного біта знайденого шаблону, і пошук продовжується. Розраховане під час тестування значення ймовірності p повинно бути не менше 0,01. В іншому випадку (p <0,01) двійкова послідовність не є абсолютно випадковою.

Тест на відповідність шаблону перекриття складається з підрахунку кількості визначених шаблонів, знайдених у вихідній послідовності. Подібно до тесту зіставлення шаблонів без перекриття у Non-Overlapping Pattern Matching, вікно довжиною m біт використовується для пошуку певного шаблону довжиною m біт. Аналогічним чином виконується і сам пошук. Якщо шаблон не знайдено, вікно зсувається на 1 біт. Єдина відмінність між цим тестом і тестом на збіг шаблонів, що не перекриваються, полягає в тому, що якщо шаблон знайдено, вікно трохи переміщується вперед, а потім пошук продовжується. Розраховане під час тестування значення ймовірності p повинно бути не менше 0,01. В іншому випадку (p <0,01) двійкова послідовність не є абсолютно випадковою.

Універсальний статистичний тест Маурера визначає кількість бітів між ідентичними шаблонами у вихідній послідовності (міра, безпосередньо пов’язана з довжиною стисненої послідовності). Мета цього тесту — побачити,

чи можна дану послідовність значно стиснути без втрати інформації. Якщо ви можете це зробити, це не зовсім випадково. Під час тестування обчислюється значення ймовірності p. Якщо p < 0,01, вихідна послідовність не вважається випадковою. В іншому випадку він вважається випадковим.

Перевірка лінійної складності базується на принципі роботи регістра зсуву лінійного зворотного зв’язку (LFSR). Перевірте, чи вхідна послідовність є достатньо складною, щоб вважатися абсолютно випадковою. Абсолютні випадкові послідовності характеризуються довгими лінійними регістрами зсуву зі зворотним зв'язком. Якщо такі регістри занадто короткі, послідовність вважається не зовсім випадковою. Під час тестування обчислюється значення ймовірності p. Якщо p < 0,01, вихідна послідовність вважається невипадковою. В іншому випадку він вважається випадковим.

Перевірка періодичності підраховує частоту всіх можливих шаблонів накладення довжиною m бітів у вихідній послідовності бітів. Визначається, чи кількість повторень 2m перекриваються закономірностей довжини біта m є приблизно такою ж, як для вхідної послідовності повністю випадкових рядків бітів. Останній, як відомо, має рівномірність. Тобто кожен шаблон довжиною m біт з'являється в послідовності з рівною ймовірністю. Якщо значення ймовірності p, обчислене під час тестування, < 0,01, то ця двійкова послідовність не є абсолютно випадковою. інакше це випадково.

Перевірка наближеної ентропії зосереджена на підрахунку частоти всіх можливих шаблонів перекриття довжиною біта m у вихідній послідовності бітів. Метою цього тесту є порівняння частот перекриття двох послідовних блоків у вихідних послідовностях довжиною m і m+1 з частотами перекриття подібних блоків у повністю випадкових послідовностях. Розраховане під час тестування значення ймовірності p повинно бути не менше 0,01. В іншому випадку (p <0,01) двійкова послідовність не є абсолютно випадковою.

Кумулятивний фінальний тест складається з максимального відхилення (від нуля) протягом будь-якого раунду. Це визначається кумулятивною сумою (-1, +1) цифр, указаних у послідовності. Метою цього тесту є визначення того, чи є

кумулятивна сума підпослідовностей, що виникають у вхідній послідовності, занадто великою або занадто малою порівняно з очікуваною поведінкою такої суми для повністю випадкової послідовності введення. Тому кумулятивну суму можна вважати довільним обходом. Для випадкових послідовностей відхилення від випадкового блукання повинно бути близьким до нуля. Для деяких типів послідовностей, які не є повністю випадковими, такі відхилення від нуля в будь-якому даному раунді є дуже значними. Якщо значення ймовірності p, обчислене під час тестування, < 0,01, то вхідна двійкова послідовність не є абсолютно випадковою. інакше це випадково.

Суть тесту на довільне відхилення полягає в підрахунку кількості циклів з рівно k відвідуваннями, довільним чином обходячи кумулятивну суму. Будь-який обхід кумулятивної суми починається з часткової суми після послідовності (0,1) і перетворює її на відповідну послідовність (-1, +1). Випадковий цикл двостороннього проходження складається з кількох кроків одиничної довжини, які виконуються випадковим чином. Крім того, цей обхід починається і закінчується в тому самому елементі. Метою цього тесту є визначення того, чи відрізняється кількість відвідувань певного стану в межах циклу від аналогічної кількості для абсолютно випадкової послідовності введення. По суті, цей тест складається з 8 тестів, які виконуються для кожного з 8 станів циклу (-4, -3, -2, -1 і +1, +2, +3, +4). Це набір. Кожен такий тест визначає ступінь випадковості вихідної послідовності за наступними правилами: Якщо значення ймовірності p, обчислене під час тестування, < 0,01, то вхідна двійкова послідовність не є абсолютно випадковою. інакше це випадково.

Інший тест на випадкове відхилення підраховує загальну кількість відвідувань певного штату шляхом випадкового округлення кумулятивної суми. Мета полягає в тому, щоб визначити відхилення від очікуваної кількості відвідувань різних станів під час випадкового сканування. В основному цей тест складається з 18 тестів, які виконуються для кожного стану (-9, -8, ..., -1 і +1, +2, ..., +9). На кожному етапі робиться висновок про випадковість вхідної

послідовності. Якщо значення ймовірності p, обчислене під час тестування, < 0,01, то вхідна двійкова послідовність не є абсолютно випадковою. інакше це випадково.

Вирішено провести тестування випадкових чисел статистичним пакетом NIST за допомогою програми написаної на мові Python[30].

Для роботи цієї програми треба встановити Python 3.6, Numpy та Scipy. Запускається програма через термінал або запустивши файл Main.py у головному каталозі. Вигляд програми зображено на рис. 5.3.

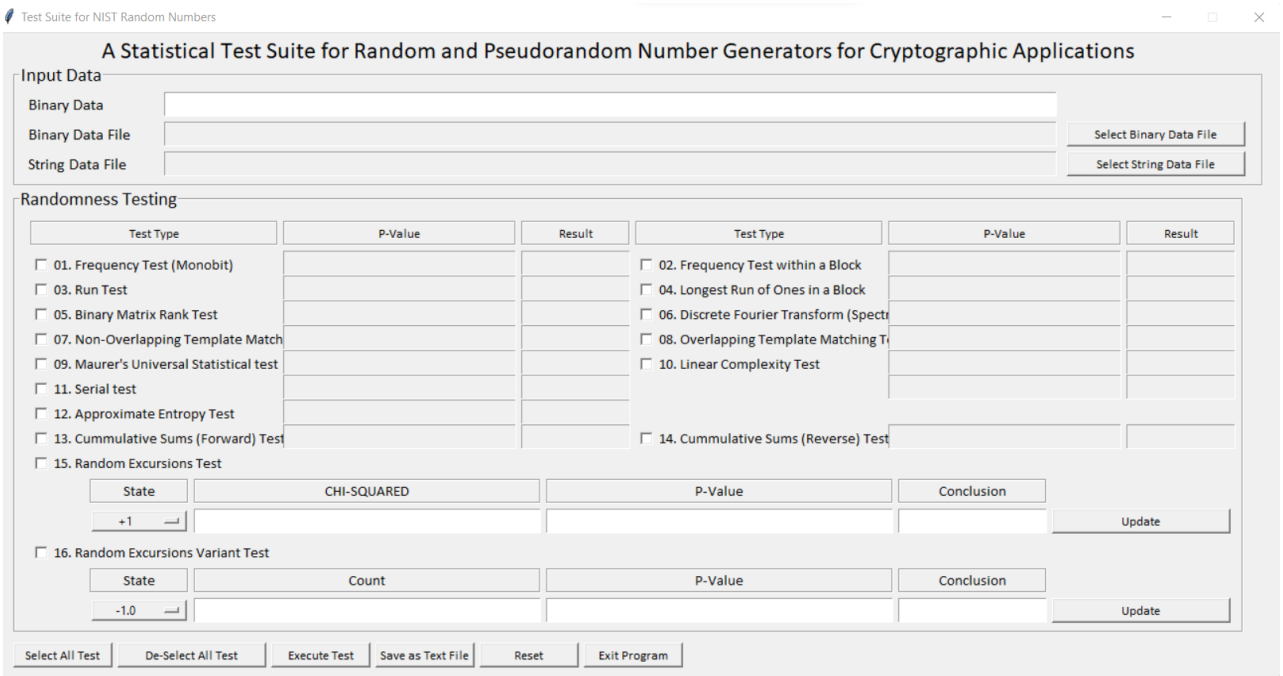

Рисунок 5.3 – Зображення програми зі статистичними тестами NIST

Вхідні дані – дані, які містять двійкові дані, файл двійкових даних і файл рядкових даних.

У поле з двійковими даними можна ввести лише двійковий рад. Наприклад, 11001001000011111101101010100010001000010110100011000010001101001100 01001100011001100010100010111000.

Щоб додати файл з двійковими даними треба відкрити діалогове вікно файлу, який буде прочитано програмою. Вибраний файл має містити лише один набір даних у двійковій формі.

Щоб додати файл з рядковими даними треба відкрити діалогове вікно файлу, у якому можна вибрати файл, який буде прочитано програмою. Вибраний файл може містити декілька наборів даних у рядковій формі.

В даній програмі є можливість вибору типу тесту, який треба виконати, натиснувши відповідний прапорець або натиснувши «Select All Test», щоб вибрати все. Щоб скасувати вибір, треба натиснути відповідний прапорець або натиснувши «De-Select All Test», щоб скасувати всі вибрані тести.

Після підготовки даних та обирання необхідних тестів треба натиснути кнопку «Execute Test», щоб виконати тест. Результат буде відображено після завершення тесту.

Щоб зберігти результати у текстовий файл, треба натиснути кнопку «Save as Text File». Відкриється діалогове вікно файлу, у якому ви зможете ввести назву файлу для результату. Кнопка «Reset» очистити усі введення та змінні. Після збереження можна натиснути кнопку «Exit», щоб вийти з програми.

Щоб отримати гарні результати, отримані числа на базі пристрою взяли по модулю 1000 та сформовано новий файл з числами (див рис. 5.4).

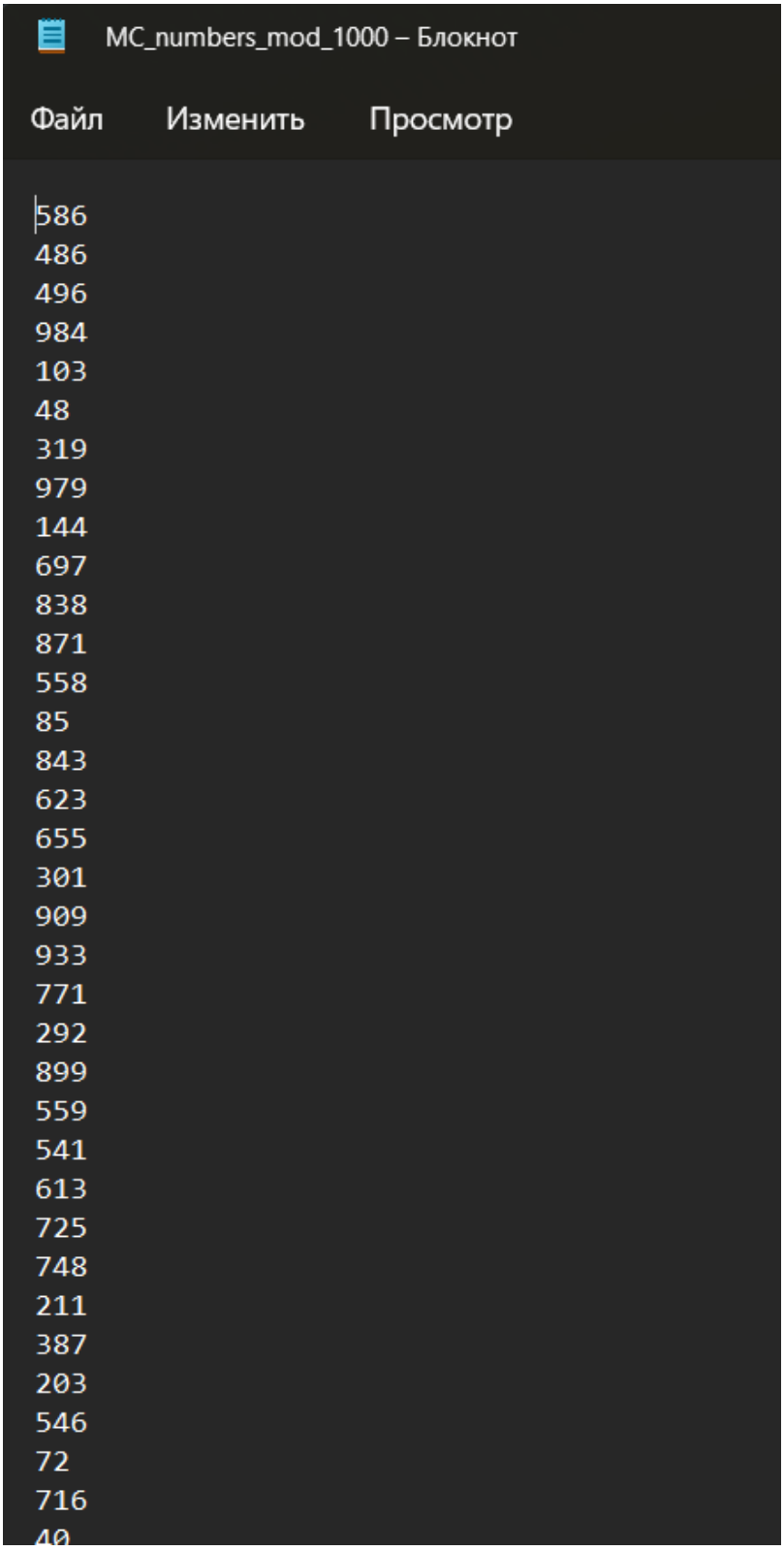

Рисунок 5.4 – Згенеровані числа по модулю 1000 у десятковому форматі

Для тестування випадкових чисел обрано файл з двійковими числами, частину якого представлена на рис. 5.5 та завантажено у програму тестування статистичними тестами NIST. Результат даної програми наведено на рис. 5.6.

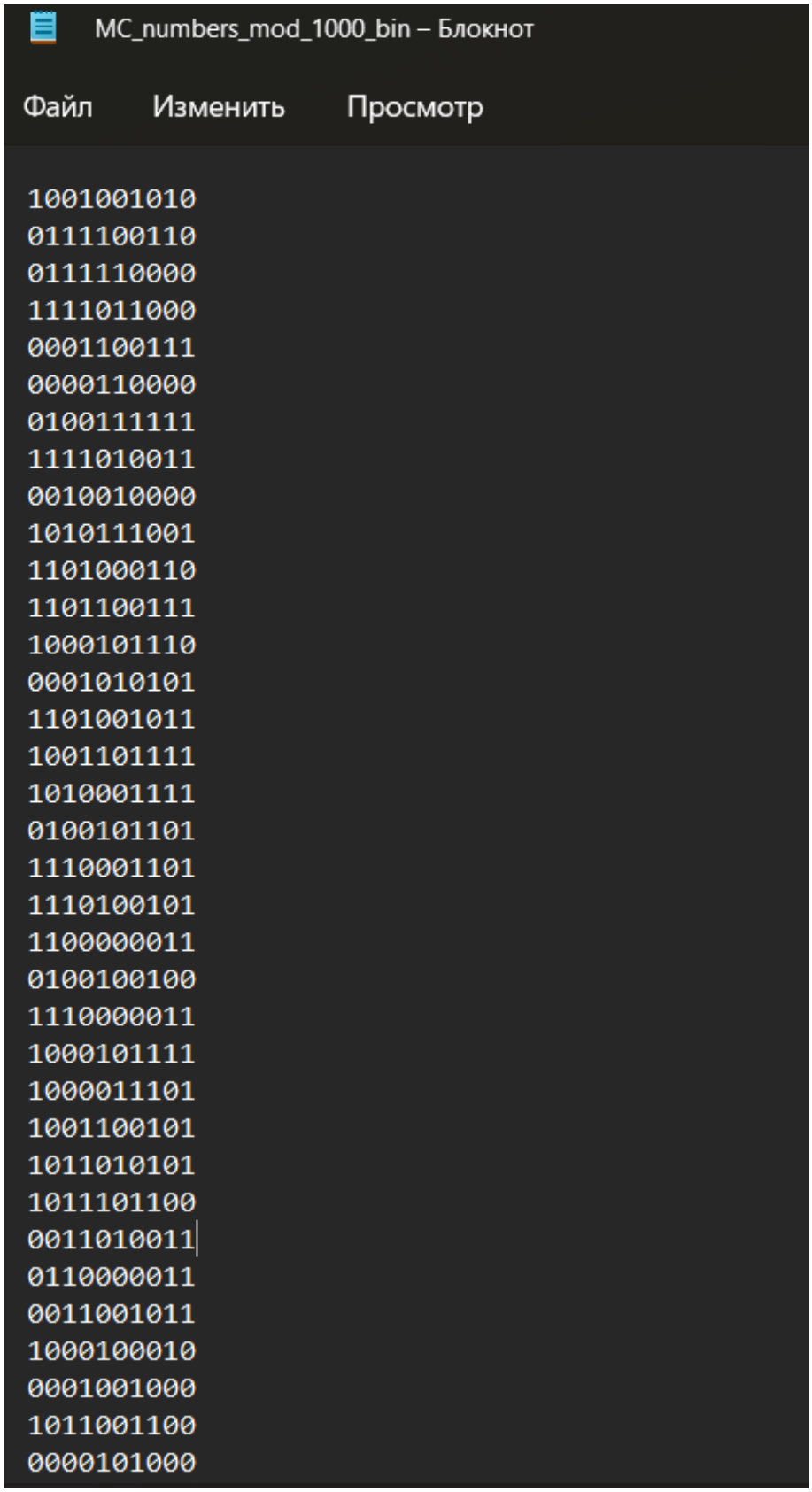

Рисунок 5.5 – Згенеровані числа по модулю 1000 у двійковому форматі

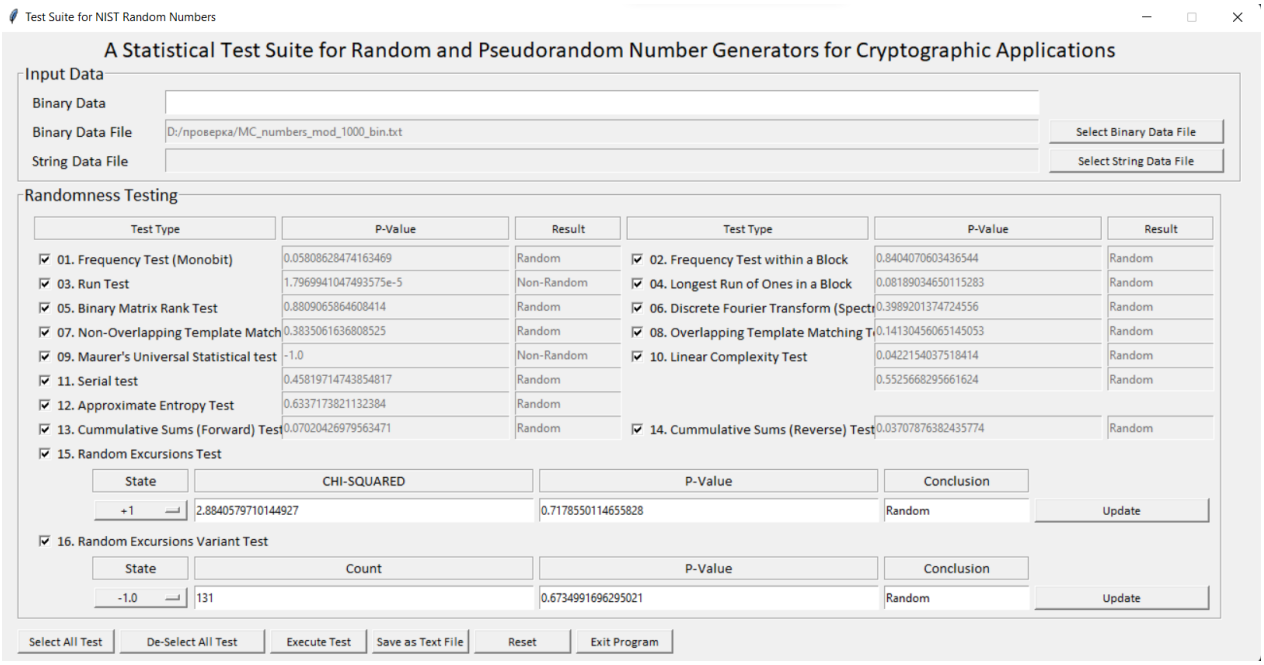

Рисунок 5.6 – Результат перевірки випадкових чисел тестами NIST

З результатів видно, що випадкові числа пройшли майже усі статистичні тести. Отримані результати наведені у. табл.5.1.

Таблиця 5.1 – Результати тестування з використанням пакету NIST

Найменування тесту	Результат тесту
Частотний побітовий тест	+
Частотний блоковий тест	+
Тест на послідовність однакових бітів	+
Тест на найдовшу послідовність одиниць у блоці	+
Тест рангів бінарних матриць	+
Спектральний тест	+
Тест на збіг шаблонів, що не перекриваються	+
Тест на збіг шаблонів, що перекриваються*	
Універсальний статистичний тест Маурера*	
Тест на лінійну складність*	
Тест на періодичність	+
Тест приблизної ентропії	+

Продовження таблиці 5.1 – Результати тестування з використанням пакету NIST

Найменування тесту	Результат тесту
Тест кумулятивних сум	+
Тест на довільні відхилення*	
Інший тест на довільні відхилення*	

«Примітка. * – Даний тест не підходить по довжині для тестування обраних даних»

У результаті дані числа пройшли 10 тестів, серед 15.

5.2 Перевірка випадкових чисел візуальним методом

Один із способів перевірити генератор випадкових чисел – створити візуалізацію чисел, які він створює. Програму для візуалізації даних обрано у роботі [31].

Щоб отримати візуалізацію за даним методом програма виконуватиме такі кроки:

а) спочатку формується сітка значень, приклад якої наведено на рис. 5.3;

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Рисунок 5.3 – Сітка візуального тесту

б) якщо число потрапляє у клітинку – наносимо на неї колір (див. рис. 5.4);

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Рисунок 5.4 – Сітка візуального тесту, коли потрапляє значення

в) якщо числа продовжують потрапляти – ставимо на цю клітинку більш глибокий відтінок кольору (див. рис. 5.5);

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Рисунок 5.5 – Сітка візуального тесту, коли на неї потрапляють числа

г) коли візуальний тест завершує роботи, ми можемо отримати остаточне зображення (див. рис. 5.6).

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Рисунок 5.6 – Результат роботи візуального тесту

Це хороший і швидкий спосіб отримати приблизне враження про якість розробленого генератора.

Наведені нижче растрові зображення генерації випадкових чисел за допомогою розробленого пристрою.

На рис. 5.7 зображено згенерована послідовність з 2000 чисел на базі мікроконтролера.

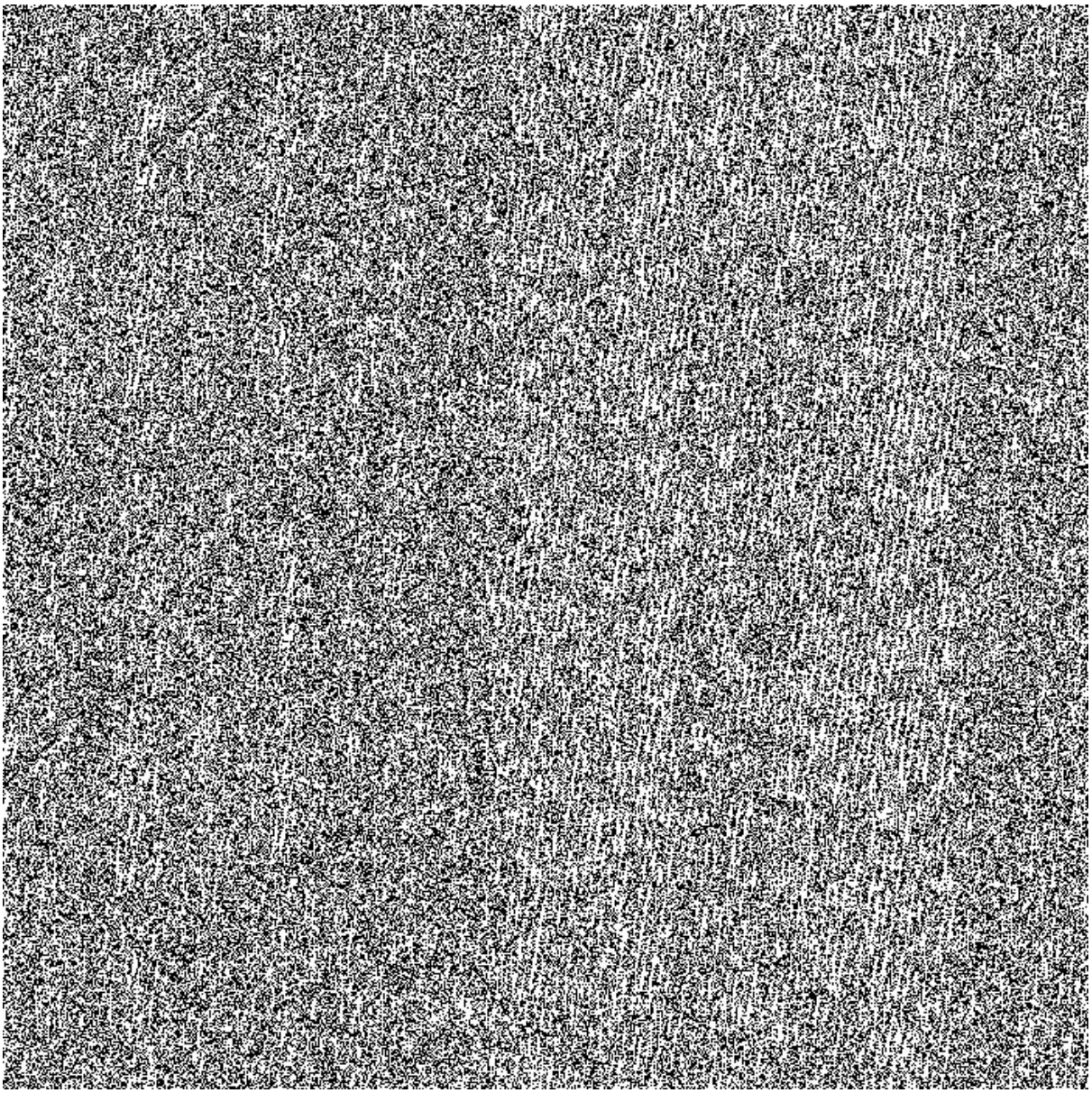

Рисунок 5.7 – Візуалізація згенерованих чесел на базі мікроконтролера

Також проведено візуалізацію згенерованої послідовності з 2000 чисел за допомогою алгоритму BBS на рис. 5.8.

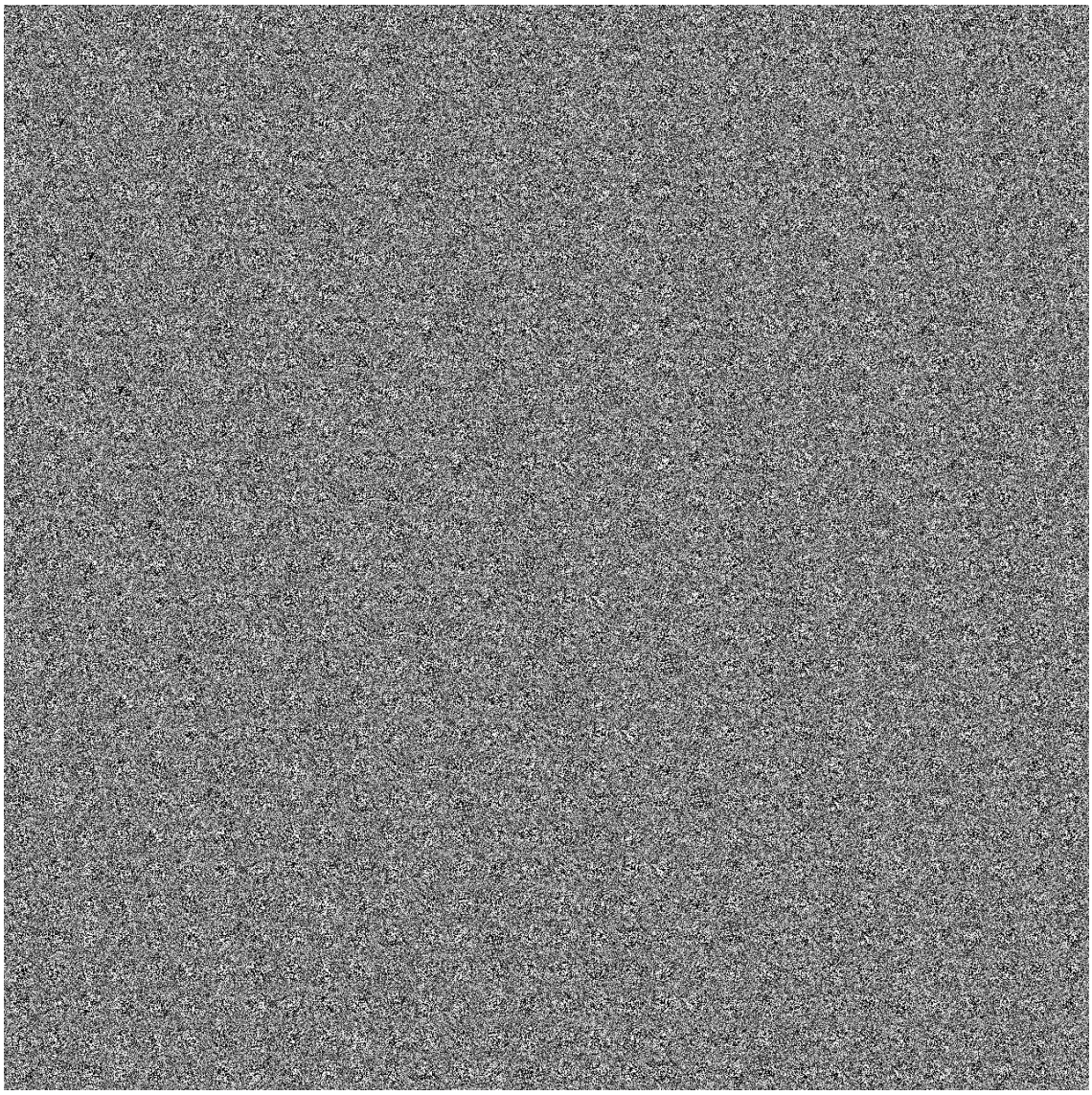

Рисунок 5.8 – Візуалізація згенерованоих чисел за допомогою алгоритму BBS

У результаті було згенеровані числа випадкові, хоча іноді вони повторюються. Це виражається у вигляді деяких смуг повторень на зображенні, але в цілому ці значення випадкові.

5.3 Порівняння характеристик випадкових чисел отриманих генераторами різних типів

Результати тестування перевірки на випадковість чисел за допомогою розробленого пристрою в даній роботі можна порівняти з результатами отриманими в роботі [30] У вказаній роботі були отримані характеристики якості випадкових чисел за допомогою магнітометру смартфону. Ці характеристики аналогічні з результатами цієї роботи.

Статистичні тести NIST пройшли такі тести: збіг шаблонів, що не перекриваються, тест кумулятивних сум, тест на лінійну складність, тест на довільні відхилення та інший тест на довільні відхилення.

Виходячи з отриманих результатів, генерація випадкових чисел на базі мікроконтролера дає гарні результати як і генерація на базі мобільного пристрою.

5.4 Висновки за розділом

Розглянуті принципи перевірки чисел на випадковість за допомогою статистичних тестів NIST та графічних тестів. За результатами проведеного тестування видно, що отримані набори чисел мають ознаки випадковості. Порівняно характеристики випадкових чисел отриманих генераторами різних типів.

ВИСНОВКИ

В роботі розроблено аппаратно-програмний комплекс генерації випадкових чисел на базі мікроконтролера та перевірено якість отриманих за його допомогою випадкових чисел.

Розглянуто основні поняття в сфері генерації випадкових чисел та основні джерела шумів, проведено їх порівняльний аналіз.

Наведено структуру комплексу та систему команд для обміну між його елементами.

Для створення пристрою генерації випадкових чисел обрано плату Arduino Mega 2560 та джерело шуму – стабілітрон КС133Г. Наведено схеми пристрою, розроблене його програмне забезпечення.

Обрано середовище, мову та засоби розробки клієнтською та серверної частин. Розроблено їх основні алгоритми та програмне забезпечення, виконане налагодження роботи.

Виконане експериментальне дослідження якості отриманих випадкових чисел статистичними та графічними тестами.

Розроблений комплекс може бути використаний на практиці для отримання випадкових чисел та у навчальному процесі при проведенні лабораторних та практичних занять.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Маслак, А. В. Генерація випадкових чисел на базі мобільних пристроїв [Текст] / А. В. Маслак, Д. О. Остапець // Сучасні інформаційні та комунікаційні технології на транспорті, в промисловості і освіті: Тези XVI Міжнародної науково-практичної конференції. – Д.: ДІІТ. – 2022. – С.149.
2. Генератор випадкових чисел [Електрон. ресурс] / Режим доступу: https://studizba.com/files/show/doc/17503-1-49557.html
3. Генератор випадкових чисел [Електрон. ресурс]/ Режим доступу: https://wiki.cuspu.edu.ua/index.php/Генератори_випадкових_чисел
4. Димакі, А. В. Апаратно-програмний генератор випадкових чисел, що сполучається з комп'ютером типу IBM PC [Текст] / А. В. Димакі, А. А. Світлаков // Вісті Томського політехнічного університету. – 2004. – №1. – С. 144-148.
5. Апаратний генератор випадкових чисел [Електрон. ресурс] / Режим доступу:https://www.wikiwand.com/ru/Аппаратный_генератор_случайных_чисел
6. Лекція. Дробовий шум [Електрон. ресурс] / Режим доступу: https://d-learn.pnu.edu.ua/data/users/4811/SR/Lections/Lection%204/Lecture%204a.pdf
7. Лекція. Шуми в операційних підсилювачах [Електрон. ресурс] / Режим доступу: https://learn.ztu.edu.ua/pluginfile.php/194897/mod_resource/content/0/%D0%A8%D1%83%D0%BC%D0%B8%20%D0%B2%20%D0%9E%D0%9F.pdf
8. Подорожний, І. В. Огляд апаратних генераторів випадкових чисел / І. В. Подорожний [Текст] / І. В. Подорожний // Молодий учений. – 2016. – № 1 (105). – С. 190-194.

9. Stipcevica, M. Quantum random number generator based on photonic emission in semiconductors [Text] / M. Stipcevica, B. Medved Rogina // Rudjer Boskovic Institute. – 2007.
10. Задков, В. Класичні та квантові генератори випадкових чисел / В. Задков, Ю. Владимирова // Суперкомп'ютери. – 2013. – №2 (14). – С. 12-21.
11. Random Sequence Generator based on Avalanche Noise [Електрон. ресурс] / Режим доступу: http://holdenc.altervista.org/avalanche/index.html
12. Vikram Belur Suresh, On-Chip True Random Number Generation in Nanometer Cmos [Text]: presented / Vikram Belur Suresh. – Massachusetts Amherst: University of Massachusetts Amherst, 2012. – 81 с.
13. Приготування однофотонних суперпозиційних станів в базисі мод Лагерра-Гауса в режимі спонтанного параметричного розсіювання [Текст] / Д.А. Турайханов та ін. – (Звістки РАН) // Серія фізична. – 2020. – №3. – С. 397-400.
14. Гріненко, Т. О., Методика вимірювання спектральної щільності потужності шуму квантової радіооптичної системи генератора випадкових чисел [Текст] / Т.О. Гріненко, О. П. Нарєжній, І. Д. Горбенко // Радіотехніка. – 2016. – №186. – С. 172-183.
15. Апаратний генератор випадкових чисел [Електрон. ресурс] / Режим доступу: https://android72.ru/allinnews/elit-odessa&com&uu/uk/generator-wikizero-aparatnij-generator-vipadkovih-cisel.html
16. Greg Taylor Behind Intel's New Random-Number Generator [Text] / Greg Taylor, George Cox // Computing / Hardware. – 2011. – С. 1-9.
17. Переваги та недоліки TCP та обмеження [Електрон. ресурс] / Режим доступу: https://www.hitechwhizz.com/2020/07/5-advantages-and-disadvantages-risks-benefitsof-tcp-protocol.html
18. Генерація криптографічно безпечної псевдовипадкової послідовності [Електрон. ресурс] / Режим доступу: https://ami.nstu.ru/~kurlaev/ib/Materials

19. Arduino mega 2560 R3 [Електрон. ресурс] / Режим доступу: https://electrobist.com/product/arduino-mega-2560-r3/
20. Arduino Mega 2560 [Електрон. ресурс] / Режим доступу: https://doc.arduino.ua/ru/hardware/Mega2560
21. Стабілітрон КС133Г [Електрон. ресурс] / Режим доступу: https://eandc.ru/catalog/detail.php?ID=10037
22. Arduino IDE software [Електрон. ресурс] / Режим доступу: https://www.arduino.cc/en/software
23. Go documentation [Електрон. ресурс] / Режим доступу:https://go.dev/doc/
24. What is Golang? Advantages and Disadvantage of Go [Електрон. ресурс] / Режим доступу: https://bestarion.com/what-is-golang/
25. GoLand [Електрон. ресурс] / Режим доступу: https://www.jetbrains.com/go/
26. Бібліотека Serial [Електрон. ресурс] / Режим доступу: https://github.com/tarm/serial
27. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications [Електрон. ресурс] / Режим доступу: https://nvlpubs.nist.gov/nistpubs/legacy/sp/
28. Набір статистичних тестів NIST [Електрон. ресурс] / Режим доступу: https://ozlib.com/1115079/informatika/nabor_statisticheskih_testov_nist
29. Генератори випадкових чисел [Електрон. ресурс] / Режим доступу: https://rudocs.exdat.com/docs/index-229485.html
30. Реалізація на Python набору статистичних тестів NIST для генераторів випадкових і псевдовипадкових чисел для криптографічних програм [Електрон. ресурс] / Режим доступу: https://github.com/stevenang/randomness_testsuite
31. Опрятний, А. О. Генерація випадкових чисел на базі мобільних пристроїв [Текст] / А. О. Опрятний, Д. О. Остапець // Сучасні інформаційні та комунікаційні технології на транспорті, в промисловості і освіті: Тези XVI Міжнародної науково-практичної конференції. – Д.: ДІІТ. – 2022. – С.150.

ДОДАТОК А

Тези доповіді на конференції

Комплекс генерації випадкових чисел на базі мікроконтролерів

Маслак А. В., Остапець Д. О.,
Український державний університет науки і технологій, Україна

У наш час досить часто застосовують прилади з використанням випадкових чисел, вибраних випадково з деякої множини. Існує декілька основних завдань, в яких використовуються випадкові числа: тестування алгоритмів, імітаційне моделювання, деякі завдання чисельного аналізу та імітація введення користувача.

Зараз для отримання випадкових чисел існує 3 основні підходи: використання таблиць випадкових чисел (числа, що є реалізацію випадкової величини); генерування випадкових чисел за допомогою деякого алгоритму (псевдовипадкові числа); використання спеціальних апаратних пристроїв(пристрої, які генерують послідовності випадкових чисел на основі виміряних параметрів з пристроїв, які хаотично видають певні шуми протікаючого фізичного процесу).

Для отримання випадкових чисел дуже часто використовують апаратний генератор випадкових чисел. Його перевагою вважається те, що випадкові числа є виміряними значеннями будь-якого випадкового фізичного процесу. До недоліків можна віднести неможливість повторного відтворення обчислень. Також є можливість отримувати непередбачувані зміни параметрів пристрою під впливом зовнішніх факторів – температури, електричних та магнітних полів та старіння елементної бази. Джерелами шумів для генератора випадкових чисел може бути: дробовий – шум в радіоелектронних пристроях (діоди, транзистори, лампи та ін.), тепловий шум – шум у пасивних елементах схеми, з яких після посилення виходить генератор випадкової напруги (напівпровідники, діоди, резистори та ін.), фазовий квантовий шум у лазерному промені – шум, який виникає завдяки фотонам, які відбиваються від поверхні (удар фотона у дзеркало), лавинний шум – під час лавинного пробою генерується струм, який складається з випадково розподілених шумових викидів (PN-переходи у режимі зворотного пробою), фазове тремтіння в кільцевих генераторах (випадкові затримки сигналу через кільцеві генератори), спонтанне параметричне розсіювання – однофотонні джерела є найважливішими квантовими елементами, що використовуються у розподілі квантових ключів (квадратично нелінійний кристал), радіоактивний розпад – джерело шуму, якому характерна випадковість кожного окремого акту розпаду (лічильник Гейгера, сцинтиляційний лічильник), атмосферний шум (радіоприймач), шум матриці фотокамери (CMOS –матриця), шум у цифровій схемі з невизначеним станом – генератора випадкових чисел із кількох інверторів, вихід кожного з яких підключений до входу іншого (інвертори з транзисторами).

В роботі проведено аналіз джерел шумів за такими характеристиками: відносна вартість реалізації, відносна складність реалізації, використання у промислових зразках та відносна швидкість генерації випадкових чисел. Виходячи з отриманих результатів одним з найкращих джерел шумів було обрано дробовий шум, який, у свою чергу, легкий та дуже дешевий у реалізації, простий у використанні. У роботі пропонується розробити комплекс генерації випадкових чисел на базі мікроконтролера Arduino Mega. Шумовий сигнал може бути отриманий за допомогою схеми з використанням стабілітрону. В роботі використано стабілітрон КС133Г, оскільки така схема може живитися від джерела 5В. Шумовий сигнал подається на вхід АЦП мікроконтролера. Отримані (оцифровані) значення амплітуди сигналу порівнюються з середнім пороговим значенням. Якщо поточне значення більше за порогове, то черговий біт випадкового числа приймається рівним 1, інакше – 0. Таким чином формується 32-розрядне випадкове число (або двійкова послідовність).

Також передбачається створити утиліту для ПК, яка буде надавати користувачу можливість отримувати випадкові числа та керувати роботою розробленого пристрою, що підключається по USB.

ДОДАТОК Б

Лістинг програми для контролера

ДОДАТОК В

Лістинг програми функції main для сервера

ДОДАТОК Г

Лістинг програми функції main для клієнта

ДОДАТОК Д

Лістинг програми логіки перевірки команди на сервері

ДОДАТОК Е

Лістинг програми логіки вибору генератора та генерації

ДОДАТОК Ж

Лістинг програми взаємодії серверу з мікроконтролером

ДОДАТОК И

Лістинг програми взаємодії серверу зі смартфоном

ДОДАТОК К

Лістинг програми реалізації алгоритма BBS

ДОДАТОК Л

Лістинг програми структури конфігурації серверу

ДОДАТОК М

Згенеровані випадкові числа