ТЕРНОПІЛЬСЬКИЙ НАЦІОНАЛЬНИЙ ЕКОНОМІЧНИЙ УНІВЕРСИТЕТ

На правах рукопису
УДК 681.325

Волинський Орест Ігорович

МЕТОДИ ПОБУДОВИ ВИСОКОПРОДУКТИВНИХ СПЕЦПРОЦЕСОРІВ НА ОСНОВІ ТЕОРЕТИКО-ЧИСЛОВОГО БАЗИСУ КРЕСТЕНСОНА

Спеціальність 05.13.05 – комп’ютерні системи та компоненти

Дисертація на здобуття наукового ступеня
кандидата технічних наук

Науковий керівник
доктор технічних наук, професор
Николайчук Ярослав Миколайович

ТЕРНОПІЛЬ – 2013

ЗМІСТ

РОЗДІЛ 3
РОЗРОБКА ТА ДОСЛІДЖЕННЯ МІЖБАЗИСНИХ ПЕРЕТВОРЕНЬ ВЕЛИКОРОЗРЯДНИХ ЧИСЕЛ У РОЗМЕЖОВАНІЙ СЗК

РОЗДІЛ 4
РОЗРОБКА ТА РЕАЛІЗАЦІЯ СПЕЦПРОЦЕСОРІВ ОПРАЦЮВАННЯ ВЕЛИКОРОЗРЯДНИХ ЧИСЕЛ У БАЗИСІ КРЕСТЕНСОНА

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ

СЗК – система залишкових класів

ТЧБ – теоретико-числовий базис

СКС – спеціалізовані комп’ютерні системи

РКС – розподілені комп’ютерні системи

ПКД – пам’ять колективного доступу

САПР – система автоматизованого проектування

R – спецпроцесор у базисі Радемахера

C – спецпроцесор у базисі Крестенсона

G – спецпроцесор у базисі Галуа

SISD (Single Instruction stream/Single Data stream) – одиночний потік команд та одиночний потік даних;

SIMD (Single Instruction stream/Multiple Data stream) – одиночний потік команд та множинний потік даних;

MISD (Multiple Instruction stream/Single Data stream) – множинний потік команд та одиночний потік даних;

MIMD (Multiple Instruction stream/Multiple Data stream) – множинний потік команд та множинний потік даних.

ПЛІС – програмована логічна інтегральна схема

ПЗП – постійний запам’ятовуючий пристрій

ОЗП – операційний запам’ятовуючий пристрій

ГІ – генератор імпульсів

АЦП – аналого-цифровий перетворювач

АЛП – арифметико-логічний пристрій

БРСЗК- бінарно-розмежована система залишкових класів

БРЗ – багатокаскадний регістр зсуву

ВСТУП

Сучасні досягнення в області мікроелектроніки, мікропроцесорної техніки та розвитку фундаментальних досліджень в теорії та розробці алгоритмів опрацювання інформаційних потоків у різних теоретико-числових базисах (ТЧБ) [1 – 5] створюють сприятливі умови для відповідного вдосконалення та покращення системних характеристик високопродуктивних процесорів [5].

При цьому успішно вирішуються задачі підвищення ефективності формування, передавання та опрацювання інформації в розподілених, вбудованих та спеціалізованих комп'ютерних системах [6].

Важливою науковою задачею при створенні та впровадженні високопродуктивних процесорів є проблемна орієнтація їх системних характеристик на виконання арифметико-логічних та модульних операцій над великорозрядними числами, а також при вирішенні задач захисту інформації в комп'ютерних мережах [7, 8]. Актуальним, також є підвищення швидкодії рішення задач теорії чисел, наприклад, пошуку найбільших спільних дільників, китайської теореми про залишки та інших, які є важливими компонентами алгоритмів криптографії, цифрового підпису, стиснення та зменшення надлишкової інформації.

У багатьох наукових школах, українських та зарубіжних, даний клас наукових та прикладних задач традиційно вирішуються на основі двійкової системи числення ТЧБ Радемахера [9 – 12], що в значній мірі обмежує функціональні можливості процесорів при зростанні алгоритмічної складності обчислювальних задач та підвищенні швидкодії опрацювання інформаційних потоків з числами в діапазоні 100-1000 і більше біт розрядності. Успішне вирішення названого класу задач може бути досягнуто за рахунок розробки математичного апарату опрацювання інформації та створення відповідної архітектури процесорів на основі ТЧБ Хаара, Крейга, Уолша, Галуа та Крестенсона [1 – 3].

Успішне застосування базису Крестенсона та системи числення залишкових класів досягнуто при реалізації високопродуктивних процесорів у системі протиповітряної оборони колишнього Радянського Союзу [14], а також в проблемно-

орієнтованих комп'ютерних системах, які були реалізовані та впроваджені в розподілених комп'ютерних системах реального часу на об'єктах нафтогазової, атомної, енергетичної та інших галузях промисловості [15].

Досвід ефективного застосування базису Крестенсона при рішенні широкого класу задач цифрового опрацювання сигналів та цифрових даних обґрунтовує значну актуальність наукової задачі створення високопродуктивних процесорів у названому базисі. Цьому сприяють сучасні можливості технологій їх проектування на основі програмованих-логічних матриць (ПЛМ), а також на мікропроцесорних платформах відомих фірм Analog Device, Dallas Semiconductor, MAXIM, Altera, Xilinx Intel, Texas Instruments, Motorola та інших [13, 14, 16, 17].

Перспективним підходом вдосконалення методів побудови високопродуктивних процесорів є використання розмежованої системи числення ТЧБ Крестенсона [18], що базується на використанні цілочисельної, нормалізованої та досконалої форм системи залишкових класів (СЗК).

У створенні та розвитку теорії та реалізації цифрових процесорів опрацювання інформації відіграли важливу роль зарубіжні та українські вчені: Stallings W., Tanenbaum A., Patterson D., Майоров С.А., Палагін О.В., Самофалов К.Г., Мельник А.О.,Тарасенко В.П, Романов В.О., Тарасов І.Є, Николайчук Я.М. [9-12, 15, 20-25]. Значний внесок у розвиток теорії побудови високопродуктивних процесорів в базисі Крестенсона зробив Акушський І.А., Брюхович Є.І., Торгашев В.А., Николайчук Я.М., Червяков Н.І., Синьков М.В. [26-31].

Актуальність теми

Розвиток теорії універсальних та спеціалізованих процесорів тісно пов'язаний з відповідним розвитком двійкової системи числення, тобто теоретико-числового базису Радемахера [31]. Сучасні досягнення у створенні високопродуктивних процесорів пов'язані з розробкою теорії паралельних обчислень, потокової та конвеєрної організації виконання програм, застосування надоперативної та асоціативної багаторівневої пам'яті, а також становлення та розробки теоретичних положень вертикальної інформаційної технології [29].

Зростаючі жорсткі вимоги до швидкодії процесорів стимулювали дослідження у застосуванні інших, відмінних від базису Радемахера, ТЧБ. Наприклад, відомі успішні застосування базису Крестенсона, математичною основою якого є розширені поля Галуа, для побудови високопродуктивних спецпроцесорів системи залишкових класів [5, 26, 30], базису Галуа, який породжує коди поля Галуа та систему числення Галуа [32-35], а також базису Уолша, який використовується при створенні комунікаційних та сигнальних процесорів у комп'ютерних мережах [2, 36]. Однак, в даних дослідженнях мало уваги приділено міжбазисним перетворенням, що особливо актуально при роботі з великорозрядними числами.

Розробка високопродуктивних процесорів опрацювання великорозрядних чисел на основі ТЧБ Крестенсона є актуальною науковою задачею, яка дозволяє вирішити завдання вдосконалення та покращення системних характеристик обчислювальних засобів, як компонентів сучасних розподілених систем, підвищення ефективності захисту інформації в комп'ютерних мережах, а також побудови швидкодіючих спецпроцесорів кореляційного, спектрального та інших застосувань опрацювання інформації.

Зв'язок роботи з науковими програмами, планами і темами.

Представлені в дисертаційній роботі дослідження виконані в рамках плану наукових досліджень на факультеті комп'ютерних інформаційних технологій кафедрою спеціалізованих комп'ютерних систем Тернопільського національного економічного університету за темами:

- «Розробка теорії та комп'ютерних засобів спеціалізованих комп'ютерних систем на основі теоретико-числових базисів Крестенсона-Галуа», державний реєстраційний номер 0106U012530;
- "Розробка теоретичних засад методів формування та цифрового опрацювання даних у розподілених спеціалізованих комп'ютерних системах", державний реєстраційний номер 0112U008458;
- НДР «Розробка алгоритмів функціонування захистів електропередач за коротких замикань на основі теорії кореляційних функцій», реєстраційний номер СКС-40-2012.

– НДР «Методи та засоби побудови безпровідних мультимедійних сенсорних мереж на основі модулярної арифметики» реєстраційний номер СКС-04-2013 «Б».

Мета і завдання дослідження. Метою роботи є вирішення наступних науково-технічних завдань:

1) систематизувати характеристики, проаналізувати сфери застосування процесорів, реалізованих у різних ТЧБ, та дослідити особливості і переваги машинної арифметики СЗК з метою визначення перспективи застосування перетворень базису Крестенсона для розв'язання задач з великою обчислювальною складністю;
2) дослідити арифметику та форми СЗК і розробити методи приведення СЗК до досконалої форми з метою спрощення часової та апаратної складності процесорів опрацювання великорозрядних чисел;
3) розробити метод виконання операції модулярного множення у розмежованій матрично-модульній системі числення;
4) розробити метод високопродуктивного перетворення чисел з базису Радемахера в базис Крестенсона на основі модулів пам'яті;
5) розробити метод швидкодіючого перетворення чисел з позиційної системи базису Радемахера в систему залишкових класів базису Крестенсона;
6) вдосконалити метод кодування, модульного сумування та множення у теоретико-числовому базисі Хаара-Крестенсона;
7) дослідити алгоритми ділення та модулярного експоненціювання у базисі Крестенсона;
8) розробити апаратні структури та дослідити характеристики компонентів спецпроцесорів у базисі Крестенсона;
9) розробити структурні та принципові рішення спецпроцесорів опрацювання великорозрядних чисел базису Крестенсона;
10) розробити та реалізувати у промисловості програмно-апаратні засоби опрацювання цифрових даних у базисі Крестенсона.

Об'єктом дослідження є процеси цифрового опрацювання великорозрядних чисел високопродуктивними спеціалізованими процесорами у базисі Радемахера-Крестенсона.

Предметом дослідження є методи, способи та засоби опрацювання великорозрядних чисел та міжбазисних перетворень Радемахера-Крестенсона.

Методи дослідження базуються на використанні теорії інформації, теорії чисел та кодування даних, теорії комп'ютерної логіки та теорії цифрового опрацювання даних. Розробка технічних засобів здійснювалась з використанням методів імітаційного моделювання автоматизованого проектування схемо- та системотехніки.

Наукова новизна одержаних результатів полягає в розвитку методів та програмно-апаратних засобів для опрацювання великорозрядних чисел на основі теоретико-числового базису Крестенсона.

Основні результати і положення, що виносяться на захист, спрямовані на створення програмно-апаратних засобів опрацювання цифрових даних у базисі Крестенсона.

1. Вперше розроблено:

- метод виконання операції модулярного множення у розмежованій матрично-модульній системі числення, який, у порівнянні з відомими, відрізняється тим, що кожен елемент матриці, який відповідає двійковому розряду 2^i, представляється кодом залишку по модулю P, а операція множення виконується шляхом додавання текучих залишків першого числа з їх подвоєними значеннями по модулю P, які відповідають одиничним елементам коду другого числа, що дозволяє зменшити обчислювальну складність модульних операцій множення та експоненціювання на 2-3 порядки;
- метод перетворення чисел з базису Радемахера в базис Крестенсона рекурентним скануванням двійкових чисел, починаючи зі старшого розряду, що, на відміну від відомих перетворень, шляхом адресної вибірки кодів залишків із модуля пам'яті, виключає операції порівняння та віднімання великорозрядних двійкових

чисел з наскрізними переносами і дозволяє підвищити швидкодію міжбазисного перетворення пропорційно розрядності двійкового числа;

- метод швидкодіючого перетворення чисел з позиційної системи базису Радемахера в систему залишкових класів базису Крестенсона, який, на відміну від відомих, шляхом бінарного розмежування, мультиплексування та рандомізації кодів залишків по модулю дозволяє максимально розпаралелити процес визначення кінцевого залишку, швидкодія якого не залежить від розрядності перетворюваних двійкових чисел.

2. Отримав подальший розвиток метод кодування, модульного сумування та множення у теоретико-числовому базисі Хаара-Крестенсона, який відрізняється від відомих представленням цифрових даних у системі взаємопростих модулів базису Крестенсона залишками в кодах базису Хаара, що дозволило замінити обчислювальні операції сумування та множення над двійковими числами матрично-модульними операціями над кодами Хаара і зменшити обчислювальну складність арифметичних операцій на два-три порядки, пропорційно розрядності кодів вхідних даних.

Практичне значення отриманих результатів

1. Розроблений пристрій визначення залишків багаторозрядного числа, який формує коди залишків розмежованої матрично-модульної системи числення у базисі Радемахера-Крестенсона, який, у порівнянні з відомими аналогами, шляхом заміни великорозрядного двійкового суматора однорозрядним повним суматором та регістрами зсуву, характеризується підвищеною швидкодією та зменшеною апаратною складністю, що розширює його функціональні можливості при опрацюванні великорозрядних чисел у задачах шифрування інформаційних потоків.

2. Розроблено спецпроцесор на основі запропонованого способу визначення залишку двійкового числа шляхом рекурентного сканування великорозрядних чисел базису Радемахера, що у порівнянні з аналогами дозволило підвищити на 2-3 порядки швидкодію визначення залишків чисел по модулю за рахунок застосування модуля пам’яті і виключити наскрізні переноси, які виникають у багаторозрядних суматорах існуючих процесорів.

3. Розроблено високопродуктивний спецпроцесор міжбазисного перетворення Радемахера-Крестенсона, що, у порівнянні з відомими аналогами, шляхом мультиплексування та рандомізації розмежованих залишків дозволило досягнути максимальної швидкодії визначення кодів залишків, у системі взаємопростих модулів, з часовою складністю перемикання двох послідовно з'єднаних логічних вентилів, незалежно від розрядності перетворюваного двійкового числа.

4. Розроблений швидкодіючий спецпроцесор кореляційного опрацювання сигналів на основі кодування, модульного сумування та множення у теоретико-числовому базисі Хаара-Крестенсона, що дозволило реалізувати однотактні модульні операції додавання та множення на вентильних матрицях і на 2-3 порядки підвищити швидкодію кореляційного опрацювання сигналів у порівнянні з аналогічними процесорами у базисі Радемахера.

Теоретичні та практичні результати роботи використано та впроваджено:

1. В Інституті мікропроцесорних систем керування об'єктами електроенергетики КД ЦІЗІТ НАН України (м.Львів) в інформаційно-діагностувальній системі моніторингу стану ізоляції в електромережах 6-35 кВ на об'єктах ВАТ ЕК «Львівобленерго» (акт від 03.12.2012р.).

2. У Тернопільському конструкторському бюро радіозв'язку «СТРІЛА» при реалізації спецпроцесорів перетворення великорозрядних чисел двійкової системи числення у систему залишкових класів (акт від 30.11.2012р.).

3. На кафедрі спеціалізованих комп'ютерних систем Тернопільського національного економічного університету при викладанні дисциплін: "Комп'ютерна логіка", "Цифрова обробка сигналів і зображень", "Теорія джерел інформації", "Спецпроцесори в різних теоретико-числових базисах" для студентів спеціальностей 8.05010203 - "Спеціалізовані комп'ютерні системи" та 8.05010201 - "Комп'ютерні системи та мережі" (акт від 18.12.2012р.).

Особистий внесок здобувача. Основний зміст роботи, наукові положення та результати сформульовано та вирішено автором самостійно. Внесок здобувача полягає в аналізі сучасного стану рішення науково-технічної задачі, розробці основних ідей, методик досліджень, структурних, принципових та алгоритмічних

рішень, організації експериментів, виготовленні дослідного взірця спецпроцесора, а також в розробці необхідного для дослідження програмного забезпечення. У публікаціях, написаних у співавторстві, здобувачеві належить:

[24] – запропонований спосіб перевірки подільності чисел на прості модулі; [37] – виконана функціональних можливосте арифметики в базисах Радемахера та Крестенсона; [67] – проаналізовано ТЧБ Радемахера та Крестенсона; [78] – запропоновано процедуру розмежування розрядної сітки процесора у базисі Радемахера-Крестенсона; [87] – запропоновано алгоритм порівняння в досконалій системі залишкових класів; [86] – запропоновано застосування розмежованої системи числення для опрацювання великорозрядних чисел; [89] – запропоновано використання матричного перемножувача базису Хаара; [90] – запропоновано структуру пристрою визначення залишку багаторозрядного числа; [91] – запропоновано алгоритм отримання залишку двійкового числа; [111] – запропоновано структуру рандомізатора міжбазисного перетворювача Радемахера-Крестенсона; [112] – запропоновано структуру матрично-модульного перемножувача в базисі Хаара; [113] – проаналізовано структуру модульних шифраторів Радемахера-Крестенсона; [114] – виконано теоретичні дослідження арифметико-логічних операцій у базисі Крестенсона; [117] – досліджено пряме перетворення системи залишкових класів; [119] – запропоновано блок-схему алгоритму пошуку залишків великорозрядних чисел Мерсена по простих модулях; [120] – проаналізовано міжбазисні перетворення Радемахера-Крестенсона; [121] – отримано аналітичні вирази оцінки швидкодії виконання арифметичних операцій в базисі Радемахера, Крестенсона та Галуа; [122] – запропоновано блок-схему мультибазисного багатофункціонального спецпроцесора.

Основні результати дисертації доповідались та обговорювались на: міжнародному симпозіумі «Питання оптимізації обчислень» (ПОО - XXXV). – Київ, 2009; 4-ій Міжнародній науково-технічній конференції Advanced Computer Systems and Networks: Design and Application (ACSN). – Львів, 2009; 10-ій міжнародній конференції „Modern problems of radioengineering, telecommunications and computer science”. – Львів-Славське, 2010; міжнародній проблемно-науковій міжгалузевій

конференції «Інформаційні проблеми комп'ютерних систем, юриспруденції, енергетики, економіки, моделювання та управління». – Бучач-Східниця, 2010; XIth International conference «The experience of designing and application of CAD systems in micro-electronics». – Lviv, 2011; міжнародній молодіжній математичній школі "Питання оптимізації обчислень (ПОО-XXXVII)" – Київ, 2011; проблемно-науковій міжгалузевій конференції «Юриспруденція та проблеми інформаційного суспільства» (ЮПІС - 2011) - Ярема, 2011; the 5-th IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS'2011) - Praga, 2011; міжнародній конференції Modern Problems of Radio Engineering, Telecommunications and Computer Science (TCSET 2012). – Львів-Славске, 2012.

Публікації. За матеріалами дисертації опубліковано 26 друкованих праць, серед яких 13 статей, з них 7 у фахових наукових виданнях (2 одноосібні), 4 патенти України на корисну модель, 9 робіт опубліковано у збірниках матеріалів конференцій.

Структура та обсяг дисертації. Дисертація складається зі вступу, чотирьох розділів, висновків та додатків (210 ст.). Основний зміст дисертаційної роботи викладений на 145 сторінках. Дисертація містить 65 рисунків, 29 таблиць та 154 посилання на джерела.

РОЗДІЛ 1
АНАЛІЗ ХАРАКТЕРИСТИК ПРОЦЕСОРІВ У РІЗНИХ ТЕОРЕТИКО-ЧИСЛОВИХ БАЗИСАХ

1.1 Систематизація характеристик та застосування процесорів, реалізованих на основі різних теоретико-числових базисів

Сучасний розвиток мікропроцесорних засобів опрацювання інформації показує широку сферу їх застосування у різних галузях науки, промисловості та суспільства. В основу методів побудови універсальних та спеціалізованих процесорів покладені фундаментальні засади здійснення алгоритмів виконання обчислювальних арифметико-логічних та комунікаційних операцій у різних теоретико-числових базисах. При цьому, широке застосування отримали наступні ТЧБ: унітарний, Хаара, Крейга, Радемахера, Крестенсона, Уолша, Галуа [1]. В останні роки при побудові проблемноорієнтованих спецпроцесорів, реалізації швидких перетворень опрацювання зображень, цифрової топографії, томографії використовуються комбінації дискретних перетворень: Хаара-Уолша [1, 2], Хаара-Радемахера, Хаара-Крестенсона [36] та інші.

Історично склалася ситуація, що реалізація процесорів універсального застосування виконано виключно у базисі Радемахера, і цей базис масово використовується практично у всіх сучасних універсальних процесорах та інформаційних системах [26].

В залежності від архітектури існує класифікація процесорів даного класу (рис.1.1) [31]. При цьому, слід відмітити, що процесори з зірково-магістральною архітектурою, як показано в роботах [5, 66, 115, 118, 123], є мультибазисними і мають структуру, показану на рис.1.2.

Функціональною перевагою мультибазисних процесорів є можливість значного підвищення швидкодії опрацювання інформаційних потоків шляхом розпаралелення та перерозподілу виконання окремих арифметико-логічних операцій, які виконуються найшвидше в різних ТЧБ. Наявність ПКД на основі асоціативної пам’яті з паралельним доступом та зірково-магістральної топології дозволяє глибоко

розпаралелити операції обміну даними між процесорами. Отже, перспектива створення високопродуктивних мультибазисних процесорів визначає актуальну наукову задачу, пов’язану з максимальним підвищенням швидкодії прямих та зворотних міжбазисних перетворень Радемахера-Крестенсона-Галуа та ін.

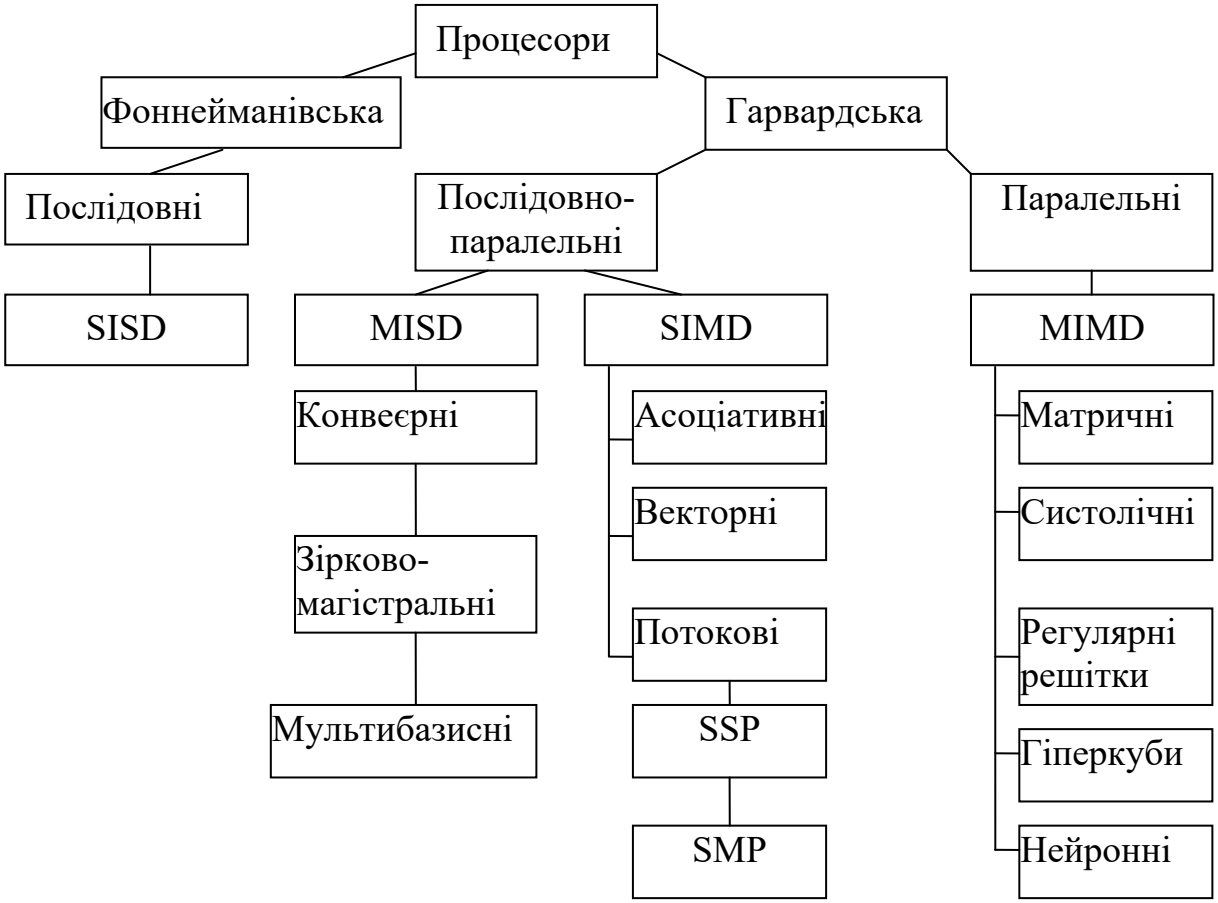
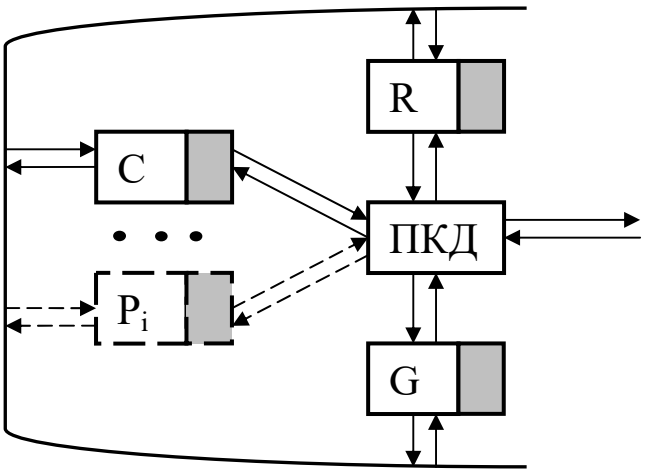


Рисунок 1.1 - Класифікація процесорів цифрового опрацювання даних



Процесорний модуль у базисі:

R – Радемахера,

C – Крестенсона,

G – Галуа;

Рисунок 1.2 - Структурна схема мультибазисного процесора

Аналіз системних характеристик, сфери застосування та архітектур процесорів, які, крім базису Радемахера, застосовують інші ТЧБ, як показано в роботі [37], дозволяє їх систематизувати по ознаках використовуваних ортогональних функцій (рис.1.3).

Рисунок 1.3 - Класифікація процесорів у різних ТЧБ та функціональні зв'язки міжбазисних перетворень

Аналіз процесорів, згідно систематизації (див. рис.1.3) [7, 12, 21, 22, 33], показує, що в ТЧБ:

- унітарному реалізовані високопродуктивні спецпроцесори кореляційного опрацювання сигналів при обчисленні структурної, модульної та еквівалентної функцій автокореляції;
- Хаара – знакова, релейна та структурна функції кореляції;
- Радемахера – коваріаційна та кореляційна функції;
- Крестенсона – коваріаційна функція;

згідно табл.1.1, де n – об'єм вибірки; $x_i, \overset{\circ}{x}_i$ - відповідно нецентрований та центрований цифровий відлік; j – дискретний зсув, M_x – математичне сподівання; D_x – дисперсія; $\overset{\cup}{Z}_{xx}$ - цифрове значення «менше з двох» x_i, x_{i+j} [5, 38-40].

Таблиця 1.1 - Дискретні кореляційні функції

Кореляційна функція	**Автокореляційна функція**
Знакова	$H_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} sign(\overset{0}{x}_i)\cdot sign(\overset{0}{x}_{i+j})\cdot$
Релейна	$B_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} \overset{0}{x}_i\cdot sign(\overset{0}{x}_{i+j})\cdot$
Коваріаційна	$K_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} x_i\cdot x_{i+j}\cdot$
Кореляційна	$R_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} \overset{0}{x}_i\cdot \overset{0}{x}_{i+j}\cdot$
Структурна	$C_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} (x_i-x_{i+j})^2\cdot$
Модульна	$G_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} \lvert x_i-x_{i+j}\rvert\cdot$
Еквівалентності	$F_{xx}(j)=\frac{1}{n}\sum_{i=1}^{n} \breve{Z}_{xx}\cdot$

Відомі реалізації процесорів у базисі Хаара при обчисленні структурної кореляційної функції [41, 42]. Перевагою базису Хаара є можливість побудови спецпроцесора з високою швидкодією шляхом аналого-цифрового перетворення вхідних даних у базис Хаара, а також високої швидкодії виконання арифметико-логічних операцій у базисі Хаара. Недоліком таких процесорів є апаратна складність, зумовлена надлишковістю кодування даних у базисі Хаара при високій точності представлення даних x_i.

Перевагою даних процесорів є регулярність архітектури та глибокий рівень розпаралелення обчислювальних процесів за рахунок того, що на інтервалі аналого-цифрового перетворення розгортуючого типу одночасно відбувається процес запису даних у багатоканальний регістр зсуву, зсув інформації в регістрі, обчислення квадратів різниць $(x_i - x_{i+j})^2$, модульних різниць $|x_i - x_{i+j}|$ та цифрових значень $\overset{\cup}{Z}_{xx}$, а

також паралельне накопичення в кожному з каналів сум обчислених значень та виконання операції ділення.

Недоліком таких процесорів є використання унітарного базису, що приводить до значної апаратної складності та низької швидкодії. Даний клас процесорів знайшов широке застосування в оптоелектроніці, розпізнаванні зображень [43, 44].

Застосування різних ТЧБ для перетворення, передавання та опрацювання інформаційних потоків в сучасних комп'ютерних системах показує, що:

1) унітарний базис найширше застосовується в інформаційно-вимірювальних системах при реалізації багатоканальних АЦП розгортуючого типу, в спецпроцесорах цифрової томографії та оптоелектронних процесорах розпізнавання зображення [43, 44];

2) базис Хаара широко використовується для реалізації процесорів швидких Вейвлет-перетворень, а також організації доступу інформації в базах даних на основі інформаційних вікон [2, 3];

3) базис Крейга [45] широко використовується в інформаційно-вимірювальних системах та око-процесорах [43];

4) в базисі Радемахера [9, 31] реалізована виключна більшість універсальних процесорів комп'ютерної та комунікаційної техніки, а також сигнальних процесорів різних застосувань [46-49];

5) найпоширеніше базис Уолша застосовується для цифрового опрацювання та стиснення зображень [50];

6) базис Крестенсона, який породжує систему числення залишкових класів, знайшов успішне застосування для побудови спецпроцесорів стиснення інформації [28], реалізації високопродуктивних процесорів опрацювання інформаційних потоків в системах протиповітряної оборони та опрацювання великорозрядних чисел в системах криптозахисту інформації [51];

7) базис Галуа, серед відомих базисів отримав, особливо широке застосування для побудови спецпроцесорів та рішення прикладних задач в системах перетворення та передавання інформації [5, 33, 83, 84];

8) аналого-цифрового перетворення та кодування інформації [52, 53];

9) передавання інформації на основі кодів Баркера та М-послідовностей [54-56];

10) стиснення інформації [57, 58].

В якості теоретико-числового базису кодування даних та реалізації процесорів провідні світові виробники (Intel, Texas Instruments, Analog Device, Motorola та інші) використовують двійкову систему числення базису Радемахера [59-64]. У табл.1.2 [31] приведені характеристики зростання тактової частоти роботи універсальних процесорів базису Радемахера, а на рис.1.4 [31] показано порівняння продуктивності процесорів та пам'яті на прикладі фірми Intel.

Таблиця 1.2 - Характеристики тактової частоти роботи універсальних процесорів базису Радемахера

Роки	1980	1985	1990	1995	2000	2005	2010
Тип процесора	8080	286	386	Pentium	PIП	P-IV	Pentium Core
Тактова частота (МГц)	1	6	20	150	750	3800	4400
Час одного такту (не)	1 000	166	50	6	1.6	0.26	0.22

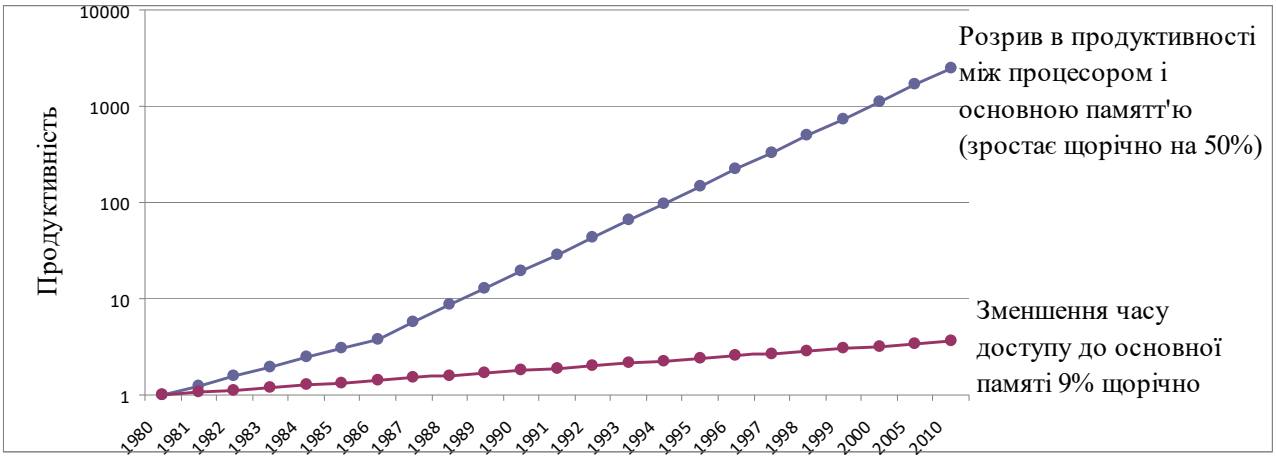


Рисунок 1.4 - Характеристики продуктивності процесорів та пам'яті базису Радемахера

Значні успіхи досягнені при побудові спецпроцесорів на основі комбінованого використання різних ТЧБ, наприклад Хаара-Крестенсона, Крестенсона-Галуа [65], а також реалізації високопродуктивних мультибазисних

RCG-процесорів на основі базисів Радемахера, Крестенсона та Галуа [66]. Оскільки обчислювальні операції модульної арифметики, які виконуються за один такт, найбільш швидкодійні у базисі Крестенсона, то поглиблення теоретичних засад реалізації арифметики базису Крестенсона є найбільш перспективним фактором вдосконалення та покращення системних характеристик широкого спектру спецпроцесорів сучасних комп'ютерних систем, що визначає високий рівень актуальності дослідження в цьому напрямку.

1.2 Теоретико-числові базиси, які породжують системи числення, та їх кодові матриці

В сучасних системах кодування та опрацювання інформації широко використовуються теоретико-числові базиси на основі дискретних кусково-постійних ортогональних функцій. До таких теоретико-числових базисів належать:

унітарний - $(Uni(n,\theta,i))$;

Хаара - $Har(n,\theta,i)$;

Крейга - $Crg(n,\theta)$;

Уолша - $Had(h,x)$;

Радемахера - $Rad(n,\theta)$;

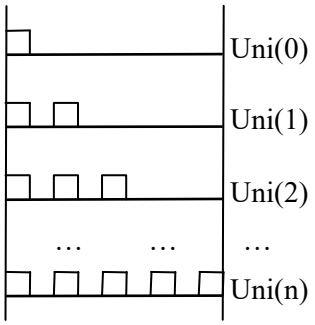
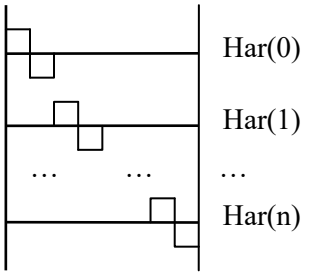
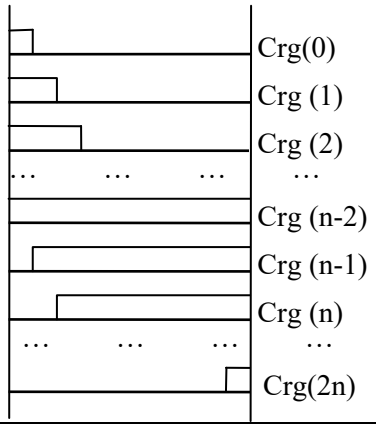
Крестенсона - Kr;

Галуа - G.

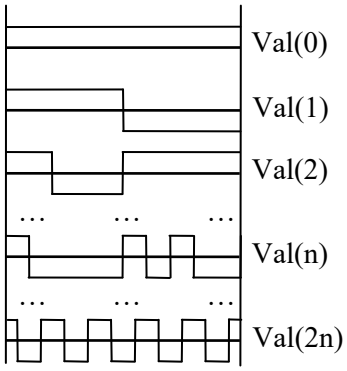
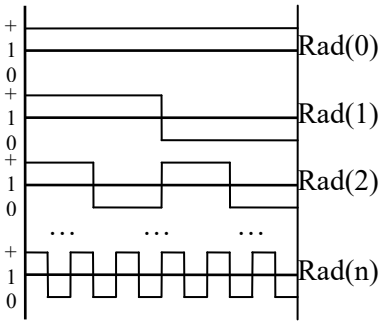
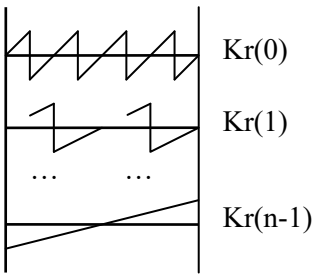
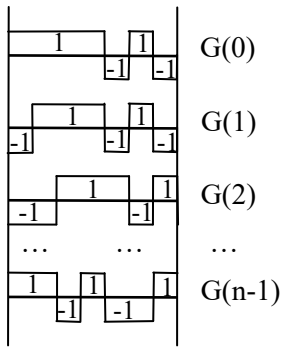
Серед названих ТЧБ системи числення породжують наступні: унітарний, Хаара, Радемахера, Крестенсона та Галуа. Оскільки ортогональні функції ТЧБ представляються у нормованому діапазоні [+1;-1], то для утворення кодової матриці відповідного базису необхідно замінити [-1] на [+1], а [+1] на [0]. У результаті такого транспонування утворюється кодова система ТЧБ у логічному Булевому базисі.

Систематизація характеристик ортогональних функцій та кодових матриць в ТЧБ, які породжують системи числення, приведена в табл.1.3 [34, 67].

Таблиця 1.3 - Систематизація характеристик та кодових матриць ТЧБ

Базис та його ортогональні функції	Базисна функція	Кодова матриця
1	2	3
Унітарний 	$Uni(n,\theta,i) = sign[\sin(2^n \pi(\theta + + i \cdot 2^{-n}))]$	$M_{Uni} = \begin{vmatrix} 0 & 0 & 0 & ... & 0 & 0 & 0 \\ 1 & 0 & 0 & ... & 0 & 0 & 0 \\ 1 & 1 & 0 & ... & 0 & 0 & 0 \\ 1 & 1 & 1 & ... & 0 & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ 1 & 1 & 1 & ... & 1 & 0 & 0 \\ 1 & 1 & 1 & ... & 1 & 1 & 0 \\ 1 & 1 & 1 & ... & 1 & 1 & 1 \end{vmatrix}$
Хаара 	$Har(n,\theta,i) = sign[\sin(i2^n \pi, \theta]$	$M_{Har} = \begin{vmatrix} 0 & 0 & 0 & ... & 0 & 0 & 0 \\ 1 & 0 & 0 & ... & 0 & 0 & 0 \\ 0 & 1 & 0 & ... & 0 & 0 & 0 \\ 0 & 0 & 1 & ... & 0 & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & 1 & 0 & 0 \\ 0 & 0 & 0 & ... & 0 & 1 & 0 \\ 0 & 0 & 0 & ... & 0 & 0 & 1 \end{vmatrix}$
Крейга 	$Crg(n,\theta) = sign[\sin((2^n - 1) \times \times \pi \cdot \theta)]$	$M_{Crej} = \begin{vmatrix} 1 & 0 & 0 & ... & 0 & 0 & 0 \\ 1 & 1 & 0 & ... & 0 & 0 & 0 \\ 1 & 1 & 1 & ... & 0 & 0 & 0 \\ ... & ... & ... & ... & ... & ... & ... \\ 1 & 1 & 1 & ... & 1 & 1 & 1 \\ 0 & 1 & 1 & ... & 1 & 1 & 1 \\ 0 & 0 & 1 & ... & 1 & 1 & 1 \\ ... & ... & ... & ... & ... & ... & ... \\ 0 & 0 & 0 & ... & 0 & 0 & 1 \end{vmatrix}$

Продовження таблиці 1.3

1	2	3
Уолша 	$Had(h,x)=\prod_{i=1}^{K}[ri(x)]hi$	$M_{Uolh}=\begin{vmatrix} 0 & \dots & 0 & 0 & 0 & 0 & 0 \\ 0 & \dots & 0 & 1 & 1 & 1 & 1 \\ 1 & \dots & 0 & 0 & 0 & 1 & 1 \\ 1 & \dots & 0 & 1 & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \dots & 0 & 0 & 1 & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & \dots & 0 & 1 & 0 & 1 & 0 \end{vmatrix}$
Радемахера 	$Rad(n,\theta)=sign[2^{n}\pi\cdot\theta]$	$M_{Rad}=\begin{vmatrix} 0 & 0 & 0 & \dots & 0 & 0 & 0 \\ 0 & 0 & 0 & \dots & 0 & 0 & 1 \\ 0 & 0 & 0 & \dots & 0 & 1 & 0 \\ 0 & 0 & 0 & \dots & 0 & 1 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 1 & 1 & \dots & 1 & 1 & \\ 1 & 1 & 1 & \dots & 1 & 1 & 1 \end{vmatrix}$
Крестенсона 	$Kr=res\sum_{i=1}^{n}(B_i\cdot b_i)\bmod P$ $b_i=resN_i(\bmod p_i)$	$M_{Cres}=\begin{vmatrix} P_1 & P_2 & \dots & P_n \\ 0 & 0 & \dots & 0 \\ 1 & 1 & \dots & 1 \\ 2 & 2 & \dots & 2 \\ 0 & 3 & \dots & 3 \\ \dots & \dots & \dots & \dots \\ b_1 & b_2 & \dots & b_n \end{vmatrix}$
Галуа 	$N_j=f(C_{j-n-1},\dots C_{j-1},C_j),$ $G=res\sum_{j=0}^{n=1}C_{j-1}\cdot A(\bmod 2)$	$M_G=\begin{vmatrix} 0 \\ 0 \\ \dots \\ 0 \\ 1 \\ \dots \\ 1 \\ 1 \end{vmatrix}$

Кожен з названих базисів характеризується визначеним об'ємом кодової матриці для представлення даних [33]. При цьому найбільш надлишковим базисом є унітарний, в якого кодова матриця $V = N^2$, а число активних кодових елементів $n = N^2/2$, де $N-$ діапазон кодування даних. Аналогічну надлишковість забезпечує базис Хаара, в два рази меншу надлишковість забезпечує базис Крейга, тобто $V = N^2/4$, а $n = N^2/8$. Максимально широке застосування для кодування даних в сучасних комп'ютерних системах (КС) отримали базиси Радемахера та Крестенсона, в яких $V = N\log_2 N$. Дані базиси, відповідно, породжують двійкову систему числення та систему числення залишкових класів [25].

Базис Уолша максимально широко використовується в сучасних телекомунікаційних КС. Даний базис породжує систему ортогональних шумоподібних сигналів, які використовуються в сотових системах мобільного зв'язку [83-86].

Найменшу надлишковість кодування даних забезпечує базис Галуа, кодова матриця якого $V = N$, а $n = N/2$.

З табл.1.3. видно, що найбільш компактні кодові матриці формують системи ортогональних функцій базисів Радемахера, Крестенсона та Галуа. При чому кожен з названих базисів породжує окрему систему числення, яка використовується для реалізації арифметики відповідних універсальних та спеціальних процесорів.

1.3 Формалізація графів мікропрограм арифметичних модулів процесорів різних ТЧБ та оцінка їх обчислювальної складності

Широке застосування базису Радемахера ґрунтується на досить простій реалізації арифметики позиційної двійкової системи числення, яка включає вісім базових операцій:

1) додавання «+»;
2) зсув «→ ←»;
3) множення «×»;
5) знакова (старшинства) «< >»;
6) віднімання «-»;
7) ділення «/»;

4) рівності «=»; 8) модульна «mod».

Важливе значення також має програмно-апаратна, структурна та обчислювальна складність міжбазисних перетворень в середовищі досліджуваних ТЧБ.

З метою підвищення ефективності міжбазисних перетворень запропонована бінарно-розмежована система числення залишкових класів (БРСЗК) [68].

Продуктивність процесорів, які реалізуються в різних ТЧБ, перш за все, визначаються аналітикою арифметико-логічних операцій та структурою арифметичного модуля.

Структурна схема арифметичного модуля в базисі Радемахера представлена на рис.1.5 [9, 75, 76].

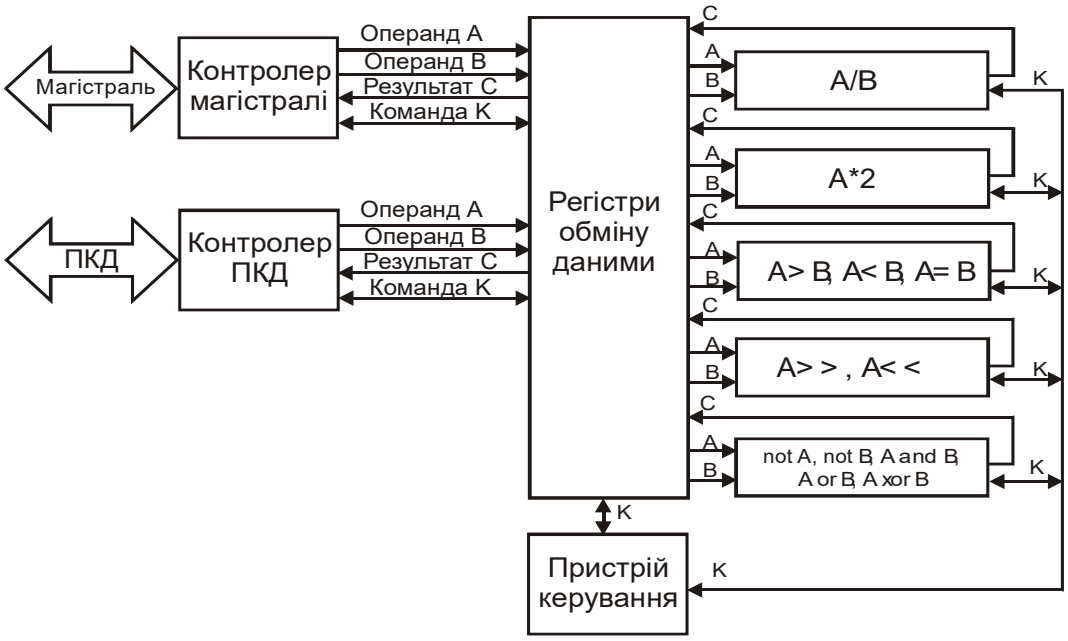


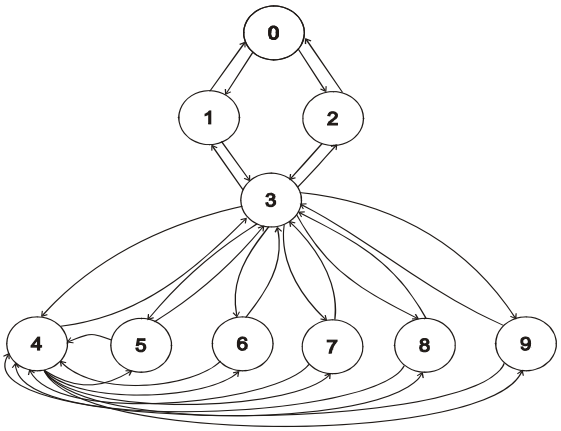
Рисунок 1.5 - Структурна схема арифметичного модуля в кодовому базисі Радемахера

До складу арифметичного модуля в кодовому базисі Радемахера входять такі функціональні елементи: пристрій керування; контролер магістралі; контролер ПКД; регістри обміну даними; блоки операцій ділення, множення на 2, порівняння, зсуву та логічних операцій; шини даних та команд.

Схема працює наступним чином: контролери ПКД та магістралі здійснюють моніторинг пакетів даних і при виявленні пакету, призначеного для арифметичного

модуля Радемахера, здійснюють його декодування і запис операндів A, B та команди K_i у відповідні регістри обміну даними. Пристрій контролю генерує сигнал активації для відповідного блоку виконання операцій і переходить у стан очікування.

Після виконання логіко-арифметичних дій над операндами A і B блок виконання операцій заносить результат C у відповідний регістр та повідомляє пристрій керування про завершення операції. Пристрій керування записує результат C в буфер контролера ПКД або магістралі в залежності від вимоги. Приклад реалізації процесу роботи модуля можна представити у вигляді графа мікропрограми (рис.1.6) [75]:



0 – Ініціалізація арифметичного модуля;
1 – Контроль та обмін даними з ПКД;
2 – Контроль та обмін даними з магістраллю;
3 – Операції регістру обміну даними;
4 – Операції пристрою керування;
5 – Арифметичне ділення;
6 – Множення на 2;
7 – Операції порівняння;
8 – Операції зсувів;
9 – Логічні операції;

Рисунок 1.6 - Граф мікропрограми функціонування арифметичного модуля в базисі Радемахера

Вершинам графу присвоєні номери 0, 1, …, 9, що відповідає кожному функціональному елементу, який виступає оператором модуля арифметичних операцій.

За допомогою даної графової моделі обчислюється тривалість виконання мікропрограм для виконання операцій ділення, множення на 2, порівняння, зсуву та логічних операцій в базисі Радемахера.

Кожній вершині присвоюється T_i - час виконання такту мікрооперації у залежності від тактової частоти процесора (0,01; 0,1; 1 ГГц) і вимірюється відповідно в мілі- , мікро- та наносекундах, згідно виразу [9]:

$$T_i = \tau_i + \vartheta_i, \qquad (1.1)$$

де τ_i, ϑ_i - відповідно інтервали часу ініціювання та виконання мікрооперації.

Якщо такт T_i має змінну тривалість, то вершина T_i позначається трьома можливими значеннями $(T_i = T_{min}, T_i = \overline{T}, T_i = \max,)$.

Тривалість виконання мікропрограми визначається сумою тривалості тактів:

$$t = \sum_{i=1}^{m} n_i T_i, \qquad (1.2)$$

де n_i - кількість звернень до і-го оператора.

Згідно (1.1) та (1.2) обчислюється час виконання мікропрограм для арифметичного модуля в базисі Радемахера (1 – процес ініціалізація арифметичного модуля, 2 – процес контролю та обміну даними з ПКД та магістраллю, 3 – процес виконання операцій регістру обміну даними, 4 – процес виконання операцій пристрою керування, 5 – процес виконання арифметичних операцій)

На діаграмі (рис. 1.7) [76] показано часові характеристики виконання мікропрограм в базисі Радемахера.

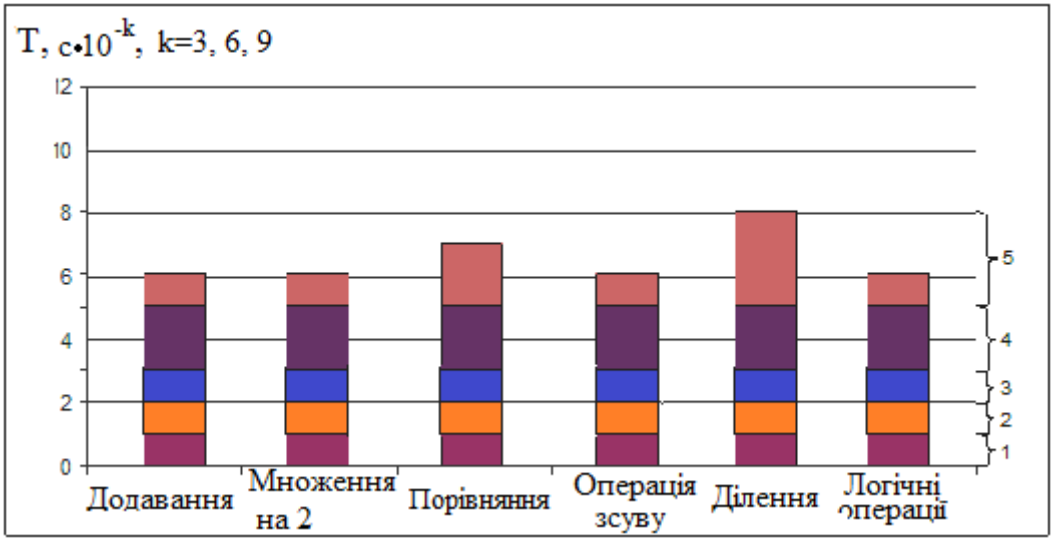


Рисунок 1.7 - Часові характеристики виконання мікропрограм арифметичного модуля в базисі Радемахера

Проведемо аналіз виконання мікропрограм для модуля в базисі Крестенсона. Даний модуль виконує тільки дві базові арифметичні операції додавання і множення, оскільки ці операції найефективніше і найпростіше реалізуються в СЗК [69-74].

Структурна схема арифметичного модуля в базисі Крестенсона представлена на рис.1.8 [69].

Рисунок 1.8 - Структурна схема арифметичного модуля в кодовому базисі Крестенсона.

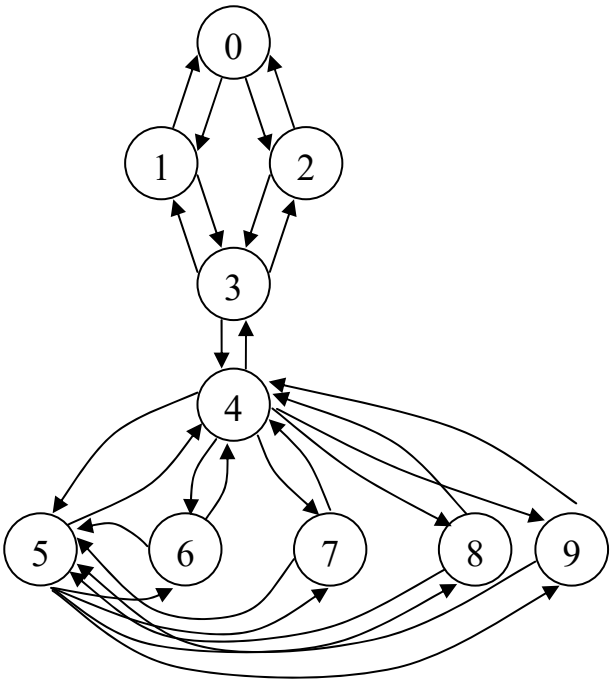
До складу арифметичного модуля в кодовому базисі Крестенсона входять такі функціональні елементи:

- пристрій керування;
- регістри обміну даними;
- контролер магістралі;
- блоки операцій сумування;
- контролер ПКД;
- коректуюча матриця множення;
- блоки міжбазисного перетворення;
- шини даних та команд.

Схема працює наступним чином: контролери ПКД та магістралі здійснюють моніторинг пакетів даних і при виявленні пакету, призначеного для арифметичного модуля Крестенсона, здійснюють його декодування, переведення операндів А і В за допомогою блоку міжбазисного перетворення в двійкове представлення базису Крестенсона і запис операндів А, В та команди $К_i$ у відповідні регістри обміну даними. Пристрій контролю генерує сигнал активації для відповідного блоку виконання операцій і переходить у стан очікування.

Після виконання логіко-арифметичних дій над операндами А і В блок виконання операцій заносить результат С у відповідний регістр та повідомляє пристрій керування про завершення операції. Пристрій керування передає результат

С на блок міжбазисного перетворення, після якого С в базисі Радемахера надходить у буфер контролера ПКД або магістралі в залежності від вимоги. Приклад реалізації процесу роботи модуля можна представити у вигляді графа мікропрограми (рис. 1.9.) [73].



0 – Ініціалізація арифметичного модуля;
1 – Контроль та обмін даними з ПКД;
2 – Контроль та обмін даними з магістраллю;
3 – Міжбазисні перетворення;
4 – Операції регістру обміну даними;
5 – Операції пристрою керування;
6 – Сумування;
7 – Множення;
8 – Подвоєння;
9 – Піднесення до степеня.

Рисунок 1.9 - Граф мікропрограми функціонування арифметичного модуля в базисі Крестенсона

Вершинам графу присвоєні номери 0, 1, …, 9, що відповідає кожному функціональному елементу, який виступає оператором модуля арифметичних операцій.

За допомогою даної графової моделі розраховується тривалість виконання мікропрограм для виконання операцій сумування та множення.

Час виконання мікропрограм для арифметичного модуля в базисі Крестенсона (1 – 6 – відповідно процеси: ініціалізації арифметичного модуля; контролю та обміну даними з ПКД та магістраллю; міжбазисного перетворення; виконання операцій регістру обміну даними; процес виконання операцій пристрою керування;

виконання арифметичних операцій) обчислюється згідно (1.1) та (1.2) (рис. 1.10.) [73].

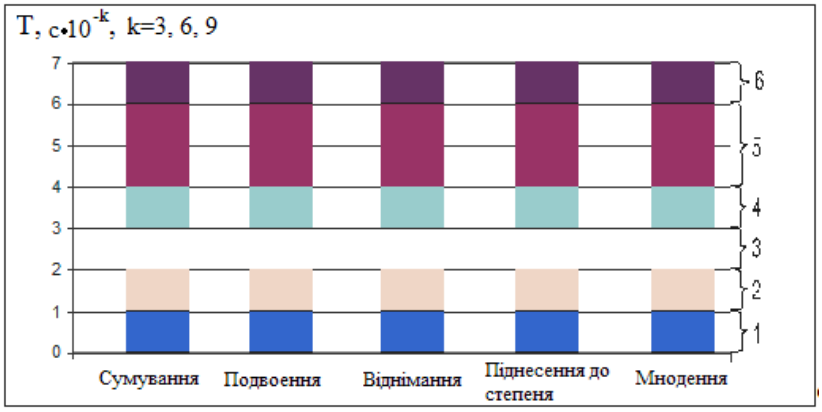


Рисунок 1.10 - Час виконання мікропрограм арифметичного модуля в базисі Крестенсона.

З рис. 1.10. можна зробити висновок, що час виконання мікропрограм для функцій множення та сумування є однаковим, що дозволяє реалізовувати складні операції для задач цифрової обробки даних.

Виконання мікропрограм для модуля в базисі Галуа, який здійснює арифметичні операції інкрементування і декрементування, що реалізуються шляхом зсуву даних в регістрі Галуа вправо або вліво згідно структурної схеми арифметичного модуля в кодовому базисі Галуа (рис.1.11) [80-82].

До складу арифметичного модуля в кодовому базисі Галуа входять такі функціональні елементи:

- пристрій керування;
- контролер магістралі;
- контролер ПКД;
- блоки міжбазисного перетворення;
- регістри обміну даними;
- блоки виконання операцій інкрементування та декрементування;
- шини даних та команд.

Схема працює наступним чином: контролери ПКД та магістралі здійснюють моніторинг пакетів даних і при виявленні пакету, призначеного для арифметичного модуля Галуа, здійснюють його декодування, переведення операндів A і B за

допомогою блоку міжбазисного перетворення в двійкове представлення базису Галуа і запис операндів А або В та команди $К_i$ у відповідні регістри обміну даними.

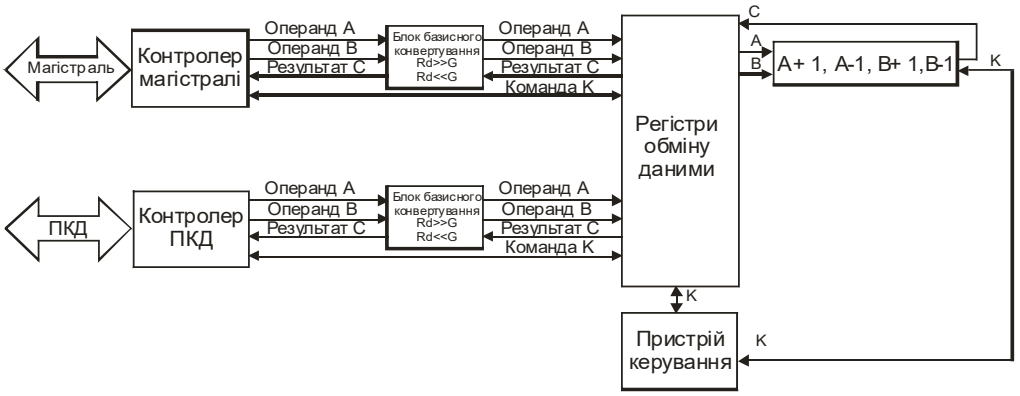
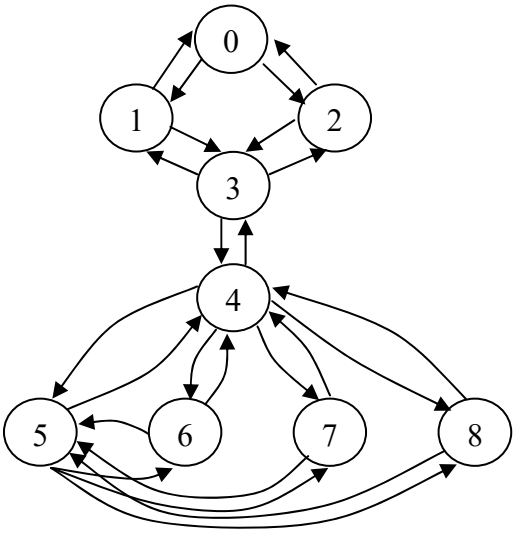


Рисунок 1.11 - Структурна схема арифметичного модуля в кодовому базисі Галуа

Після виконання логіко-арифметичних дій над операндами А або В, аналогічно модулям в інших ТЧБ, в блок виконання операцій заносить результат С у відповідний регістр та повідомляє пристрій керування про завершення операції. Пристрій керування передає результат С на блок базисного перетворення, після якого С в базисі Радемахера надходить у буфер контролера ПКД або магістралі в залежності від вимоги. Приклад реалізації процесу роботи модуля представляється у вигляді графа мікропрограми (рис.1.12) [73]:



0 – Ініціалізація арифметичного модуля;
1 – Контроль та обмін даними з ПКД;
2 – Контроль та обмін даними з магістраллю;
3 – Міжбазисні перетворення;
4 –8 – Операції: регістру обміну даними (4); пристрою керування (5); додавання (6); множення (7); інкрементування/декрементування (8).

Рисунок 1.12 - Граф мікропрограми функціонування арифметичного модуля в базисі Галуа.

Вершинам графу присвоєні номери 0, 1, …, 8, що відповідає кожному функціональному елементу, який виступає оператором модуля арифметичних операцій.

За допомогою даної графової моделі розраховується тривалість виконання мікропрограм для рішення операцій сумування та множення.

Часові затрати роботи мікропрограм виконання арифметичних операцій модулями в базисі Галуа обчислюється згідно (1.1) та (1.2) (1 – 6 – відповідно процеси: ініціалізації арифметичного модуля; контролю та обміну даними з ПКД та магістраллю; міжбазисного перетворення; виконання операцій регістру обміну даними; виконання операцій пристрою керування; виконання арифметичних операцій додавання, множення, інкрементування / декрементування) (рис. 1.13.).

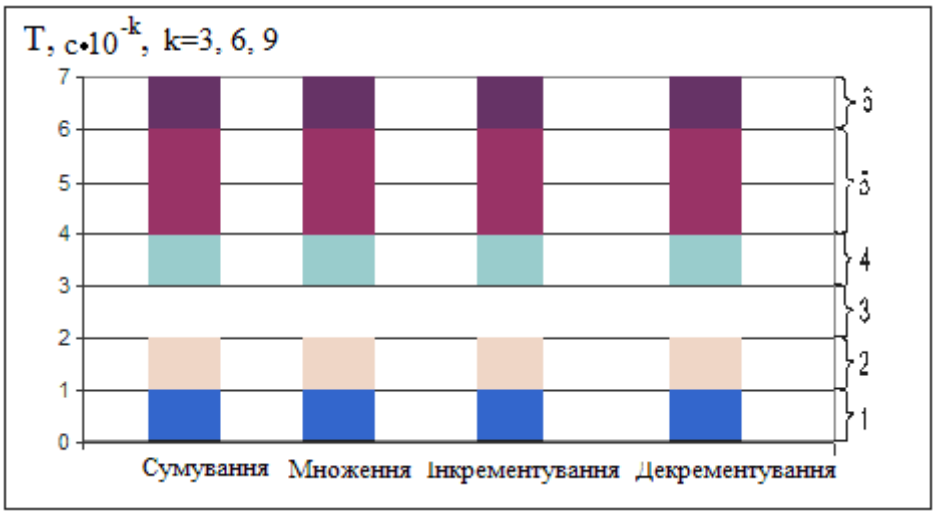


Рисунок 1.13 - Час виконання мікропрограм арифметичного модуля в базисі Галуа.

В той же час, як показано в роботі [77], операція додавання та множення в базисі Галуа потребує не менше двох тактів роботи процесора, що приводить до більш низької швидкодії даного класу процесорів по відношенню до процесорів у базисі Крестенсона.

Виконаний аналіз реалізації базисних функцій мультибазисних RCG-процесорів з позицій створення високопродуктивних спецпроцесорів, орієнтованих на опрацювання великорозрядних чисел, дозволяє зробити висновок, що доцільний розвиток теорії ТЧБ Крестенсона з метою реалізації всіх базових функцій

арифметики системи залишкових класів є перспективною та актуальною науково-технічною задачею.

1.4 Теоретичні засади виконання та характеристики арифметико-логічних операцій у базисах Радемахера та Крестенсона

Система числення залишкових класів, яка породжується теоретико-числовим базисом Крестенсона, характеризується суттєвими перевагами по відношенню до базису Радемахера при виконанні операцій додавання та множення. Оскільки СЗК непозиційна і в ній відсутні наскрізні переноси при виконанні арифметичних операцій, то процес виконання операцій додавання та множення виконується за один такт на основі матриць [28, 78]. Тому є важливим реалізація та дослідження міжбазисних перетворень Радемахера-Крестенсона для проектування спецпроцесорів опрацювання великорозрядних чисел.

Двійкова система числення належить до позиційних систем числення і є частковим випадком системи числення залишкових класів з набором модулів p^i, де p=2, 3,…, 8, 10, 16…; i=0, 1, 2,… [25, 31].

Арифметичні операції над двома числами у двійковій системі числення базису Радемахера описуються наступними виразами:

$$X=\sum_{i=0}^{n-1} x_i 2^i,\ x_i \in \overline{0,1}; Y=\sum_{i=0}^{n-1} y_i 2^i,\ y_i \in \overline{0,1}.$$

Тобто, двійкові коди чисел X і У :

$$X=(x_{n-1},x_{n-2},...,x_i,...,x_0); Y=(y_{n-1},y_{n-2},...,y_i,...,y_0).$$

визначаються на основі модульних операцій згідно аналітичних виразів:

$$X \rightarrow \begin{cases} resX(\bmod 2^0)=x_0; \\ resX(\bmod 2^1)=x_1; \\ resX(\bmod 2^i)=x_i; \\ resX(\bmod 2^{n-1})=x_{n-1}; \end{cases} \qquad Y \rightarrow \begin{cases} resY(\bmod 2^0)=y_0; \\ resY(\bmod 2^1)=y_1; \\ resY(\bmod 2^i)=y_i; \\ resY(\bmod 2^{n-1})=y_{n-1}, \end{cases}$$

де *res*- операція знаходження найменшого невід'ємного залишка по модулю 2^i.

Невиконання умови взаємної простоти модулів в різних розрядах двійкових кодів відповідно ускладнює алгоритми додавання та множення двійкових чисел, оскільки при виконанні операції додавання між двійковими розрядами виникають наскрізні переноси з молодших в старші розряди

$$\begin{array}{cccccccc} & x_{n-1} & \dots & x_i & \dots & x_1 & x_0 \\ + & y_{n-1} & \dots & y_i & \dots & y_1 & y_0 \\ \hline P_n & \leftarrow P_{n-1} & \leftarrow \dots & \leftarrow P_i & \leftarrow \dots & \leftarrow P_1 & \leftarrow \\ \downarrow & \downarrow & & \downarrow & & \downarrow & \downarrow \\ S_n & S_{n-1} & \dots & S_i & \dots & S_1 & S_0 \end{array} .$$

Наявність наскрізних переносів при виконанні операції додавання в базисі Радемахера в 2n-разів знижує швидкодію виконання операції сумування чисел по відношенню до тактової частоти роботи процесорів. У зв'язку з цим існують різні способи побудови суматорів базису Радемахера з більш швидкою реалізацією наскрізних переносів, що особливо важливо при виконанні операцій над великорозрядними числами [31].

Відсутність взаємної простоти модулів системи числення базису Радемахера обумовлює складну складність алгоритмів множення двійкових чисел згідно графа (рис.1.14) [31] , де AND-лінійка операторів, яка формує n n-розрядних результатів логічного множення множеного X на розряди множника Y, які зсуваються праворуч на R_i *(i=1, 2,..., n-1*).

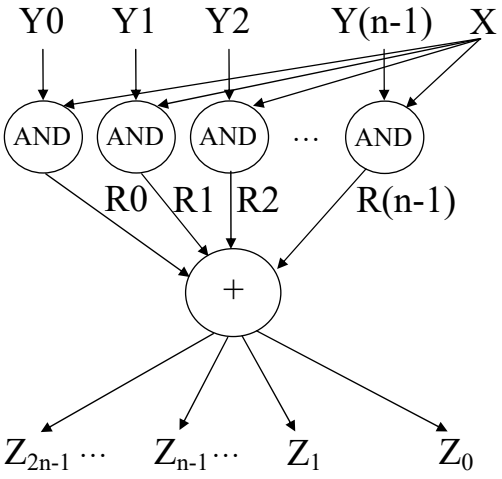


Рисунок 1.14 - Граф виконання операції множення в базисі Радемахера

Теоретичною основою системи числення залишкових класів (СЗК) є Китайська теорема про залишки [79], на основі якої реалізується пряме та зворотнє перетворення СЗК:

$$X \rightarrow \begin{cases} resX(\bmod p_0) = b_0; \\ resX(\bmod p_1) = b_1; \\ resX(\bmod p_i) = b_i; \\ resX(\bmod p_k) = b_k; \end{cases} \qquad Y \rightarrow \begin{cases} resY(\bmod p_0) = a_0; \\ resY(\bmod p_1) = a_1; \\ resY(\bmod p_i) = a_i; \\ \cdots \quad \cdots \quad \cdots \\ resY(\bmod p_k) = a_k; \end{cases}$$

$$X = res\sum_{i=0}^{k-1} b_i \cdot B_i(\bmod P); \quad Y = res\sum_{i=0}^{k-1} a_i \cdot B_i(\bmod P),$$

де $P = \prod_{i=0}^{k-1} p_i$; p_1, p_2, ..., p_i, ..., p_k - система взаємно простих модулів;

$B_i = \frac{P}{p_i} \cdot m_i \equiv 1(\bmod\ \ p_i)$; $0 \le m_i \le p_i - 1$ нормуючі коефіцієнти базисних чисел B_i.

Виконання умови взаємної простоти модулів СЗК базису Крестенсона суттєво спрощує алгоритми виконання операцій додавання та множення над числами, представленими кодами СЗК $X = (b_0, b_1, ..., b_j, ..., b_{k-1})$ та $Y = (a_0, a_1, ..., a_j, ..., a_{k-1})$ згідно граф-алгоритмів рис.1.15, де (+)res відповідає операції $C_j = res(b_j + a_j) \bmod P_j$, а (×)res – операції $\gamma_j = res(b_j \cdot a_j) \bmod P_j$:

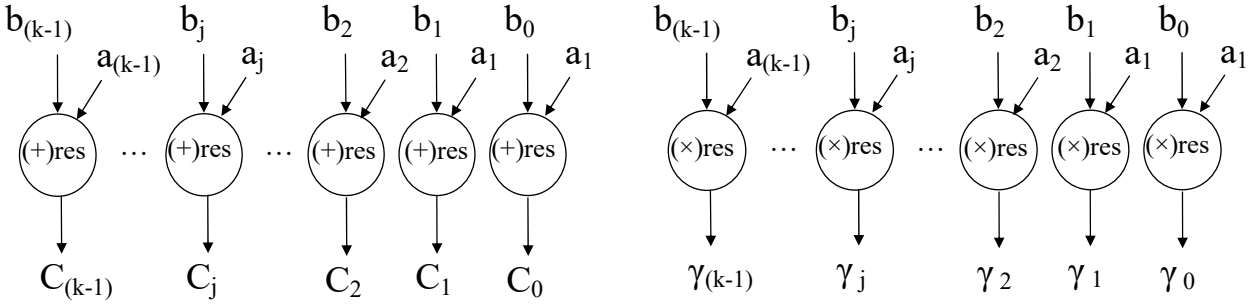


Рисунок 1.15 - Графи виконання операцій додавання та множення в базисі Крестенсона.

Порівняльна оцінка функціональних можливостей досліджуваних ТЧБ подана в табл.1.4, де *n* – розрядність процесора, *v* – час спрацьовування логічного елемента,

- - відсутність операції, де l - число модулів, добуток яких представляє дільник, k – число модулів виконання операції у відповідному ТЧБ.

Таблиця 1.4 - Функціональні можливості досліджуваних ТЧБ.

Базові операції	Базиси	
	Рademахера	Крестенсона
Додавання	$2nv$	v
Зсув	v	-
Множення	$2v(2n+1)$	v
Рівності	v	v
Знакова(старшинства)	nv	$2nvk$
Віднімання	$(3n+5)v$	v
Ділення	n^2v	$(l+1)\cdot v$
Модульна	n^2v	$2nv$

Проаналізувавши функціональні можливості ТЧБ (див. таблицю 1.4) і побудувавши графіки часових затрат (рис.1.16, 1.17, 1.18) видно, що базис Крестенсона має значні переваги у швидкості їх здійснення.

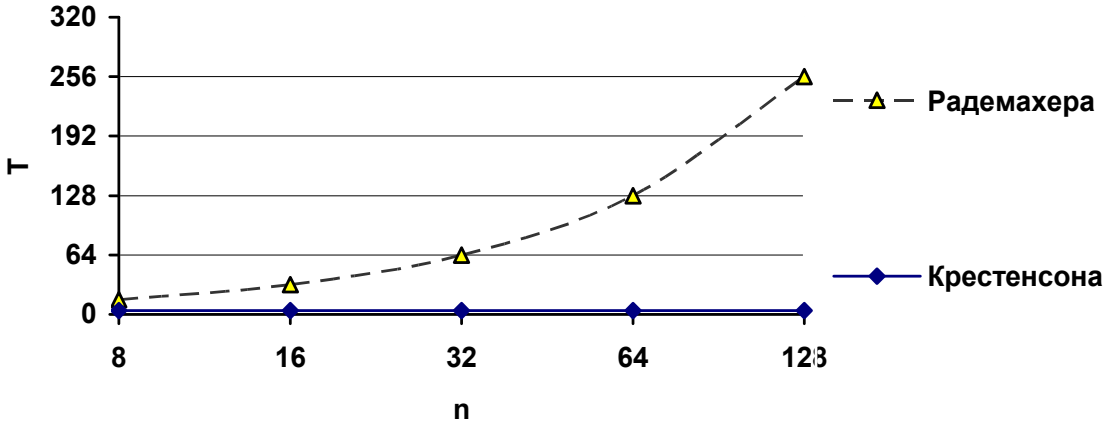


Рисунок 1.16 - Часові затрати виконання операції додавання в базисах Рademахера та Крестенсона

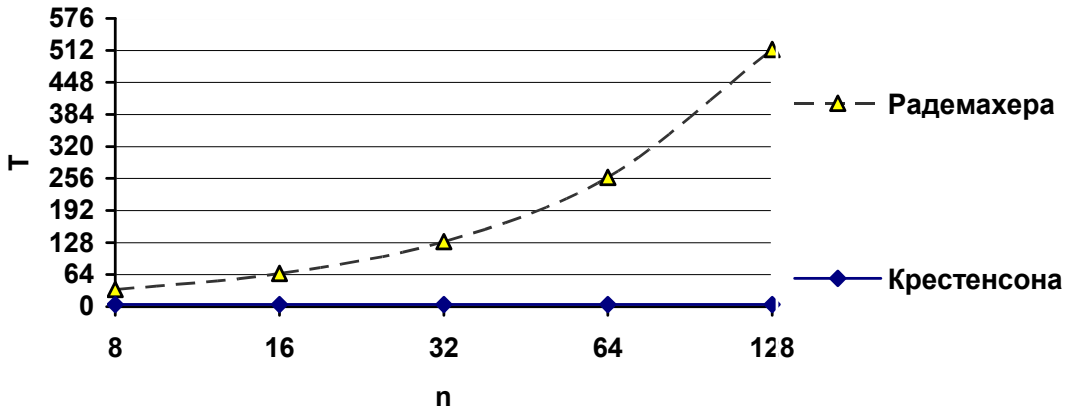


Рисунок 1.17 - Часові затрати виконання операції множення
в базисах Радемахера та Крестенсона

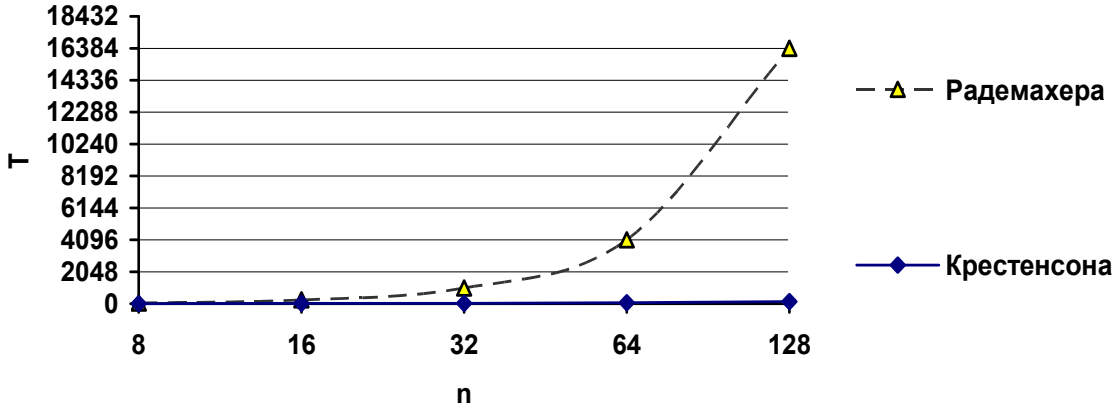


Рисунок 1.18 - Часові затрати виконання модульної операції
в базисах Радемахера та Крестенсона

Відсутність наскрізних переносів між розрядами в операціях додавання та множення в СЗК базису Крестенсона забезпечує високу швидкодію їх алгоритмів, які виконуються за один такт незалежно від числа модулів та розрядності кодів чисел СЗК. Така властивість СЗК особливо позитивна та перспективна для реалізації спецпроцесорів опрацювання великорозрядних чисел в задачах криптографії та оптимізації обчислень. В той же час, необхідність виконання міжбазисних перетворень Радемахера-Крестенсона та Крестенсона-Радемахера визначають актуальність розвитку теорії та схемотехніки спецпроцесорів таких міжбазисних перетворень.

1.5 Перспективи застосування систем числення базису Крестенсона для побудови високопродуктивних спецпроцесорів опрацювання великорозрядних чисел, постановка задачі дослідження.

Одним з перспективних шляхів вдосконалення спецпроцесорів опрацювання великорозрядних чисел є розвиток теорії арифметичних операцій непозиційної системи числення на основі базису Крестенсона – СЗК. Представлення даних в СЗК дає змогу здійснювати розпаралелення арифметико-логічного опрацювання інформації. Проте, найвагомішою перевагою СЗК перед іншими позиційними системами числення є простота однотактного виконання арифметичних операцій при відсутності наскрізних переносів. Це спрощує принципи побудови спецпроцесорів та дозволяє підвищити на 1-2 порядки швидкодію виконання операцій додавання та множення у порівнянні з позиційними системами числення.

Існує три типи основних перетворень СЗК :

- цілочисельне $N_k = res \sum_{i=1}^{k} b_i \cdot B_i (\mathrm{mod}\ P)$ [25];

- нормалізоване $[N_k]_0 = res \sum_{i=1}^{k} [b_i]_0 \cdot m_i (\mathrm{mod}\, 1)$ [5];

- досконале $[N_k]_0 = res \sum_{i=1}^{k} [b_i]_0 (\mathrm{mod} 1)$ [5],

які дозволяють реалізувати спецпроцесори базису Крестенсона з різними характеристиками апаратної та часової складності.

Крім того, в даний час активно розвивається теорія нормалізованої досконалої СЗК (НДСЗК) [78] та розмежованої СЗК (РСЗК) [24], які дозволяють успішно вирішити широкий клас задач опрацювання великорозрядних чисел у галузі криптографії та побудови високопродуктивних мультибазисних спецпроцесорів.

Однією з найважливіших операцій у всіх асиметричних алгоритмах шифрування інформаційних потоків є операція модулярного множення чисел a та b. Для підвищення ефективності обчислення $a \cdot b (\mathrm{mod}\, p)$ багатьма авторами запропоновано різні підходи, алгоритми, але в основному з використанням десяткової системи числення.

Тому розробка ефективного алгоритму з використанням теоретико-числового базису Крестенсона для зменшення обчислювальної складності формування, збільшення швидкодії на основі матричних моделей та розробка спеціалізованих програмно-апаратних засобів є актуальною задачею.

Операції модулярного множення в базисі Радемахера лежать в основі алгоритмів електронного цифрового підпису, криптографічних протоколів, розв'язування задач обчислювальної, прикладної та дискретної математики [93], визначення стійкості еліптичних кривих методом пошуку їх порядку за допомогою алгоритму Шуфа [94] тощо. Існуючі алгоритми (стандартний, швидкого множення, Blakey, Монтгомері, бінарні і т.д.) характеризуються значною обчислювальною складністю. Алгоритм модулярного множення в базисі Крестенсона за допомогою матричних обчислень дозволяє зменшити обчислювальну складність.

Проведені дослідження в підрозділі 1.3 дозволили встановити, що виконання базових операцій додавання і множення у СЗК ТЧБ Крестенсона характеризується на 1-2 порядки меншою часовою складністю по відношенню до двійкової системи числення базису Радемахера. Це створює перспективу ефективного застосування процесорів у базисі Крестенсона для зменшення обчислювальної, часової та апаратної складності спецпроцесорів в задачах, що потребують опрацювання великорозрядних чисел криптографії, дискретної математики, кореляційного та спектрального аналізу та інш., в яких використовуються модулярні операції додавання, множення та експоненціювання.

Оскільки реалізація обчислювальних операцій в універсальних комп'ютерах та контролерах найчастіше виконується в базисі Радемахера, то постає питання в прямому переході з базису Крестенсона в базис Радемахера і навпаки, що ефективно виконується за допомогою принципової розмежованої форми системи числення в базисах Радемахера-Крестенсона [95].

На основі проведених досліджень в роботі [28] показано, що використання ТЧБ Крестенсона дозволяє реалізувати алгоритм операції множення у базисі Радемахера, який має квадратичну часову складність $O1(n) = n^2$, операцією додавання з логарифмічною складністю:

$$O2(n) = \begin{cases} (\log_2 n)^2, \textit{якщо\_} n < 256 \\ n \cdot (\log_2 n), \textit{в\_інших\_випадках} \end{cases}.$$

Результати дослідження ефективності даного алгоритму шляхом емуляції на універсальному процесорі наведені на рис.1.19 [95].

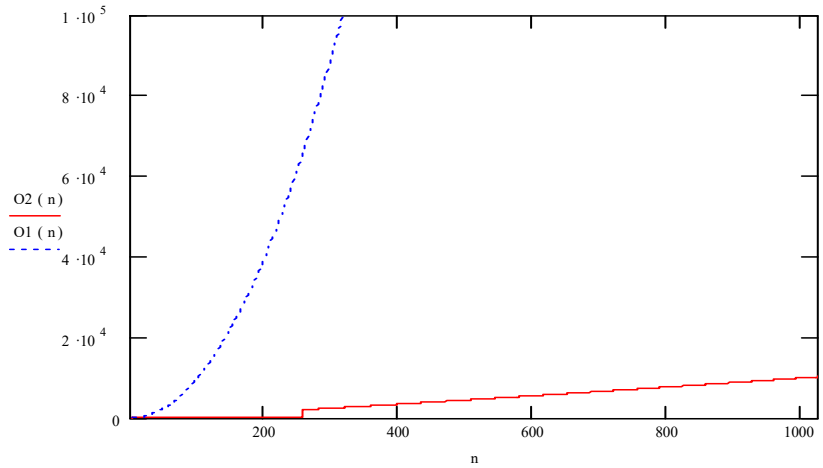


Рисунок 1.19 - Складність операції модулярного множення

Проведені експериментальні дослідження показують, що із зростанням розрядності вхідних параметрів операції модулярного множення від 0 до 256 бітів часова складність даного алгоритму на основі використання ТЧБ Крестенсона логарифмічна, а в проміжку від 256 до 1024 біт є лінійно-логарифмічною, за рахунок заміни операції множення в двохвимірному базисі Радемахера однотактною операцією додавання над розмежованими фрагментами великорозрядних чисел.

Дослідження показали, що алгоритм характеризується високою швидкодією та ефективністю виконання операції піднесення до степеня двійкового числа будь-якої розрядності за модулем *p*. Слід зазначити, що при збільшенні розрядності чисел зменшується ефективність (див.рис. 1.19), бо частина ресурсу комп'ютера буде задіяна на розв'язок службової інформації, що значно збільшує складність і кількість операцій та зменшує швидкодію.

З рис.1.19 видно, що при зростанні розрядності чисел алгоритму модульного множення відомий метод не може бути фізично реалізований з використанням сучасної мікропроцесорної техніки, особливо з врахуванням наступної операції піднесення до степеня. При цьому використання базису Крестенсона дозволяє

реалізувати можливості суттєвого підвищення захисту інформаційних потоків від несанкціонованого доступу шляхом збільшення розрядності шифрокодів [28].

Використання розпаралелення виконання арифметичних операцій над розмежованими великорозрядними числами на основі перетворень залишкових класів [78] дозволяє спростити реалізацію міжбазисних перетворень Радемахера-Крестенсона і здійснити реалізацію високопродуктивних спецпроцесорів розв'язання досліджуваних задач.

Таким чином, наукова задача розробки та реалізації високопродуктивних спецпроцесорів опрацювання великорозрядних чисел у базисі Крестенсона потребує вирішення наступних завдань:

- систематизувати характеристики спецпроцесорів реалізованих у різних ТЧБ;
- розробити теоретичні основи виконання арифметичних операцій в нормалізованій та досконалій системі залишкових класів;
- дослідити арифметику і форми СЗК з метою спрощення часової та апаратної складності алгоритмів опрацювання великорозрядних чисел;
- розробити високопродуктивні однотактні матрично-модульні операції на основі розмежованої системи числення Радемахера-Крестенсона;
- розробити методи та способи високопродуктивного перетворення чисел з базису Радемахера в базис Крестенсона;
- вдосконалити методи перетворення чисел з базису Крестенсона в базис Радемахера на основі цілочисельної та нормалізованої досконалої форми системи залишкових класів;
- оптимізувати характеристики компонентів спецпроцесорів базису Крестенсона згідно критеріїв мінімальної часової та апаратної складності;
- розробити високопродуктивні спецпроцесори визначення залишків великорозрядних чисел та однотактний спецпроцесор міжбазисного перетворення Радемахера-Крестенсона;
- реалізувати на сучасній мікроелектронній елементній базі високопродуктивні спецпроцесори кореляційного опрацювання сигналів у базисі Крестенсона та

шифрування великорозрядних чисел на основі дискретного логарифму для задач криптографії.

Теоретичне обґрунтування виконання арифметико-логічних операцій у різних формах системи залишкових класів, дослідження та реалізація спецпроцесорів міжбазисних перетворень та опрацювання великорозрядних чисел у базисі Радемахера-Крестенсона є предметом дослідження даної дисертації.

ВИСНОВКИ ДО РОЗДІЛУ 1

1. Виконана систематизація процесорів цифрового опрацювання даних на основі різних архітектур та сфери застосувань в широкому класі задач цифрового опрацювання даних на основі використання різних теоретико-числових базисів, що дозволило обґрунтувати перспективу розробки та реалізації високопродуктивних спецпроцесорів опрацювання великорозрядних чисел у системі числення залишкових класів базису Крестенсона.

2. Отримані аналітичні вирази та виконано порівняння арифметико-логічних операцій процесорів, реалізованих в теоретико-числових базисах Радемахера та Крестенсона, що дозволило встановити підвищення на 1-2 порядки швидкодії процесорів базису Крестенсона при виконанні модульних операцій додавання та множення незалежно від розрядності опрацьовуваних чисел.

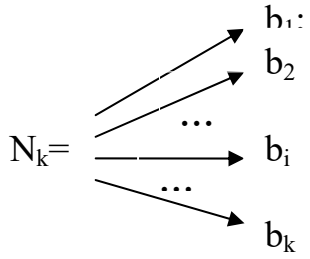
3. Проаналізовані характеристики структур та побудовані графи мікропрограм функціонування арифметичних модулів в базисах Радемахера, Крестенсона та Галуа. Побудовані діаграми часової складності виконання арифметико-логічних операцій процесорів різних ТЧБ, що дозволило обґрунтувати перспективу застосування процесорів Крестенсона при виконанні операцій додавання та множення з швидкодією на 2-3 порядки вищою у порівнянні з процесорами Радемахера.

4. Виконаний аналіз характеристик арифметико-логічних операцій у базисах Радемахера та Крестенсона, здійснено порівняння функціональних можливостей процесорів та побудовані діаграми часових затрат виконання базових операцій, що дозволило встановити основні переваги процесорів Крестенсона при вирішенні задач шифрування інформаційних потоків у сучасних комп’ютерних системах та сформулювати завдання і постановку задачі дисертаційного дослідження.

РОЗДІЛ 2
РОЗРОБКА МЕТОДІВ, ДОСЛІДЖЕННЯ АРИФМЕТИЧНИХ, МАТРИЧНО-МОДУЛЬНИХ ТА МІЖБАЗИСНИХ ОПЕРАЦІЙ У РОЗМЕЖОВАНІЙ СИСТЕМІ БАЗИСУ КРЕСТЕНСОНА

2.1 Дослідження арифметики цілочисельної форми системи залишкових класів базису Крестенсона

В основу цілочисельного перетворення СЗК покладена Китайська теорема про залишки [79]. Суть прямого перетворення цілочисельної форми СЗК полягає в тому, що, згідно з теоремами про залишки, будь-яке ціле число можна однозначно перетворити набором найменших невід'ємних залишків в системі взаємно простих модулів



$$b_i = \mathrm{res}N_k (\mathrm{mod}\, p_i), \qquad (2.1)$$

що відповідає рішенню діофантового рівняння

$$N_k = b_i (\mathrm{mod}\, p_i), \qquad (2.2)$$

яке відповідає цілочисельному рішенню лінійного рівняння:

$$N_k = a_i p_i + b_i, \qquad (2.3)$$

де a_i – ранг; b_i – найменший невід'ємний залишок.

При цьому діапазон кодування чисел N_k дорівнює:

$$P = \prod_{i=1}^{k} p_i \; ; \; 0 \le N_k \le P-1. \qquad (2.4)$$

Таким чином, ціле число N_k однозначно представляється набором залишків b_i.

Зворотне перетворення цілочисельної форми СЗК виконується згідно аналітичного виразу [25]

$$N_k = res \sum_{i=1}^{k} b_i \cdot B_i (\text{mod } P), \tag{2.5}$$

де B_i – базисні числа СЗК, які обчислюються згідно діофантового рівняння:

$$B_i = \frac{P}{p_i} \cdot m_i \equiv 1 (\text{mod } P_i). \tag{2.6}$$

Тобто, для виконання зворотнього перетворення цілочисельної форми СЗК необхідно задати параметри:

1. Набір взаємно простих модулів $p_1, p_2, ..., p_i, ..., p_k$;

2. Діапазон кодування чисел P;

3. Набір базисних чисел $B_1, B_2, ..., B_i, ..., B_к$;

4. Набір коефіцієнтів, які забезпечують ортогональність перетворень $m_1, m_2, …, m_i, …, m_k$;

5. Аналітичні вирази прямого і зворотнього перетворення:

$$b_i = res N_k (\text{mod} p_i),$$

$$N_k = res \sum_{i=1}^{k} b_i * B_i (\text{mod} P). \tag{2.7}$$

У таблиці 2.1 показаний приклад кодування чисел в цілочисельній формі СЗК, для наступної системи модулів:

$p_1=2, p_2=5$;

P=2*5=10;

$0 \le N_k \le 9$.

$$B_1 = \frac{10}{2} \cdot m_1 = 1(\text{mod}\, 2), m_1 = 1; B_1 = 5;$$

$$B_2 = \frac{10}{5} \cdot m_2 = 1(\text{mod}\, 5), m_2 = 3; B_2 = 6.$$

Таблиця 2.1 - Кодування чисел в цілочисельній формі СЗК.

N_k	b_1	b_2
0	0	0
1	1	1
2	0	2
3	1	3
4	0	4
5	1	0
6	0	1
7	1	2
8	0	3
9	1	4

Теорія виконання алгоритму операцій додавання, віднімання та множення в цілочисельній формі СЗК детально викладена в роботі [68]. В основу

арифметики залишкових класів покладено глибоке розпаралелення обробки даних, яке виконується по кожному модулю p_i окремо з виключенням міжрозрядних переносів.

Алгоритми визначення операції додавання, віднімання і множення в СЗК виконується на основі наступних математичних виразів:

1) Сумування $x_k+y_k \le P-1$:

$$x_k=(a_1, a_2, \dots a_i, \dots a_k) \quad + \quad x_k+y_k= (a_1, a_2, \dots a_i, \dots a_k) \quad (2.8)$$

$$y_k=(b_1, b_2, \dots b_i, \dots b_k) \quad \underline{(b_1, b_2, \dots b_i, \dots b_k)} \ ,$$

$$(c_1, c_2, \dots c_i, \dots c_k)$$

де $c_i=res(a_i+b_i) \bmod p_i$;

2) Сумування $x_k+y_k \ge P$:

$$x_k+y_k= (a_1, a_2, \dots a_i, \dots a_k); \quad (2.9)$$

$$+ \ \underline{(b_1, b_2, \dots b_i, \dots b_k)} \ ,$$

$$(d_1, d_2, \dots d_i, \dots d_k)$$

де $d_i=res(a_i+b_i) \bmod P$;

3) Віднімання ($x_k \ge y_k$):

$$x_k-y_k= (a_1, a_2, \dots a_i, \dots a_k) \quad (2.10)$$

$$- \ \underline{(b_1, b_2, \dots b_i, \dots b_k)} \ ,$$

$$(e_1, e_2, \dots e_i, \dots e_k)$$

де $e_i=res(a_i-b_i) \bmod P_i$;

4) Віднімання ($x_k<y_k$) $x_k-y_k=(x_k-y_k)=P+x_k-y_k$:

$$x_k-y_k= (a_1, a_2, \dots a_i, \dots a_k) \quad (2.11)$$

$$- \ \underline{(b_1, b_2, \dots b_i, \dots b_k)},$$

$$(f_1, f_2, \dots f, \dots f_k);$$

де $f_i=res(a_i-b_i) \bmod P_i$;

5) Множення ($x_k \cdot y_k$):

$$x_k * y_k = \underset{Nik}{(a_1, a_2, \dots a_i, \dots a_k)} \quad (2.12)$$

$$\underline{(b_1, b_2, \dots b_i, \dots b_k)},$$

$$(j_1, j_2, \dots j, \dots j_k)$$

де $j_i=res(a_i \cdot b_i) \bmod P_i$.

5) Ділення (x_k/y_k).

Створення повнофункціональних процесорів в базисі Крестенсона потребує вдосконалення методів реалізації операції ділення над числами з заданими наборами залишків.

Запропонований алгоритм виконання операції ділення [119] на основі доповнення в класах діофантових рівнянь, над великорозрядними числами, заданими в базисі Радемахера, що має вигляд:

$$Z = X \diagup Y,$$

в якому $X_{(2)} = (x_{n-1},...,x_i,...,x_0); x_i \in \overline{0,1}; Y_{(2)} = (y_{n-1},...,y_i,...,y_0); y_i \in \overline{0,1}$,

$$Z_{(2)} = \underbrace{(z_{r-1},...,z_i,...,z_0)}_{Z_0} + \underbrace{(b_{r-1},...,b_i,...,y_0)}_{b_0}; y_i \in \overline{0,1};$$

де Z_0 - ціла частина (ранг);

b_0 - дробова частина (залишок).

Рішення цієї задачі відповідає рівнянню в цілих числах:

$$X = Z_0 \cdot Y + b_0;\ X^* = Z_0 \cdot Y,$$

яке відповідає порівнянню:

$$X \equiv b_0 (\mathrm{mod}\, Y). \qquad (2.13)$$

Застосувавши принцип доповнення відносно рівняння (2.13) в класі діофантових рівнянь, отримаємо:

$$X^* + b_0 \equiv 0 (\mathrm{mod}\, Y).$$

Тобто, якщо Y можна представити p_j, яке входить в склад системи взаємно простих модулів базису Крестенсона, метод ділення реалізується згідно наступного алгоритму:

1) До ділимого X додаємо доповнення b_0 по модулю Y.

2) Спрощуємо систему взаємопростих модулів СЗК базису Крестенсона, які представлені простими числами $p_1, p_2,..., p_j,..., p_k$, вилучивши, при цьому, з нього модуль Y.

3) Визначаємо діапазон зміни ділимого у границях: $0 \le X \le P_j$, де $P = \prod_{j=1}^{k} p_j$. Розглянемо рішення задачі коли $Y = p_j$.

4) На основі представлення чисел в розмежованій СЗК [89] перетворюємо двійкові числа X та Y у базис Крестенсона з набором модулів $\{p_j, Y, j \in \overline{1,k}\}$ на основі матричних алгоритмів (рис.2.1).

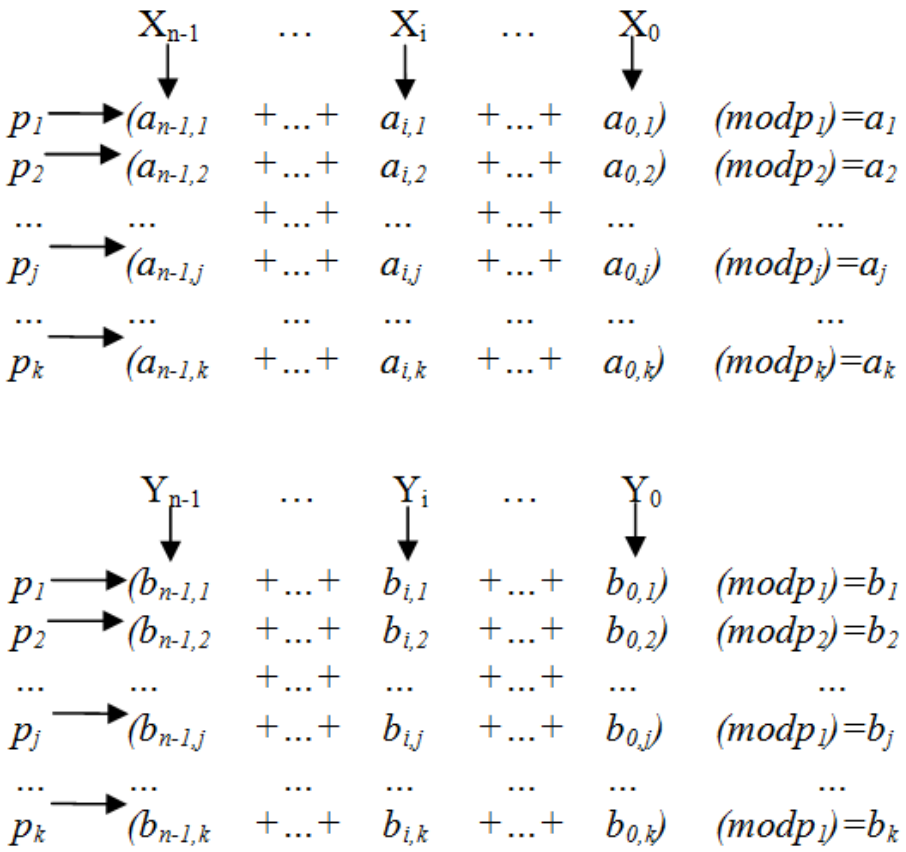


Рисунок 2.1 - Математичні алгоритми міжбазисного перетворення Радемахера-Крестенсона на основі розмежованої СЗК, відповідно числа X та Y.

5) До числа X, представленого в СЗК, додаємо числове значення доповнюючого залишку b_j^* по модулю p_j, тобто

$$+\begin{array}{c} X = (b_1,...,b_j,...,b_k) \\ b_j^* = (b_1^*,...,b_j^*,...,b_k^*) \\ \hline X^* = (x_1^*,...,x_j^*,...,x_k^*) \end{array}.$$

6) На основі алгоритму модульного множення чисел в базисі Крестенсона вирішуємо задачу знаходження інформативних залишків ділимого, згідно наступного матричного алгоритму:

$$\begin{array}{l} \times \begin{array}{l} Y = (y_1,...,y_j,...,y_k) \\ Z^* = (z_1^*,...,z_j^*,...,z_k^*) \end{array} \\ \hline X^* = (x_1^*,...,x_j^*,...,x_k^*) \end{array}.$$

Тобто $(y_{i\neq j} \cdot z_{i\neq j}^*) \bmod p_{i\neq j} = x_{i\neq j}^*$, що означає видалення неінформативного залишку по модулю p_j.

7) З представлення СЗК вилучаються неінформативні модулі із залишками y_i, що дорівнюють нулю. В результаті формуємо нову СЗК на основі інформативних $y_i \neq y_j$, тобто $Z^* = (z_1^*,...,z_i^*,...,z_l^*); i \in \overline{1,l}$, у системі модулів $p_1,..,p_i,...,p_l$. При цьому, $\prod_{i=1}^{l} p_i \geq X_{\max}$.

8) Для отримання правильного цілого результату ділення виконується відніманням від інформативних залишків СЗК згідно виразу:

$$\begin{array}{l} - \begin{array}{l} Z^* = (z_1^*,...,z_i^*,...,z_l^*) \\ \quad (1\ ,...,1\ ,...,1\ ) \end{array} \\ \hline Z\ \ = (z_1,...,z_i,...,z_l) \end{array}.$$

Таким чином, результат операції ділення отримаємо у вигляді:

$$X \diagup Y = \{Z_i\} + \{b_i\}; i \in \overline{1,l}; j \in \overline{1,k}.$$

Проведені дослідження розробленого методу виконання операції ділення в СЗК дозволяють оцінити часову складність процесора ділення в базисі Крестенсона на основі виразу

$$\tau_{div} = (l+1) \cdot \nu,$$

де l - число модулів, добуток яких представляє дільник Y.

Однією із переваг цього методу є відсутність операції порівняння в алгоритмі ділення, що зменшує часову складність операції ділення в СЗК та дозволяє розпаралелити її виконання в бінарно-розмежованій СЗК.

На рис.2.2 показані порівняльні характеристики швидкодії виконання арифметико-логічних операцій в базисі Радемахера та Крестенсона.

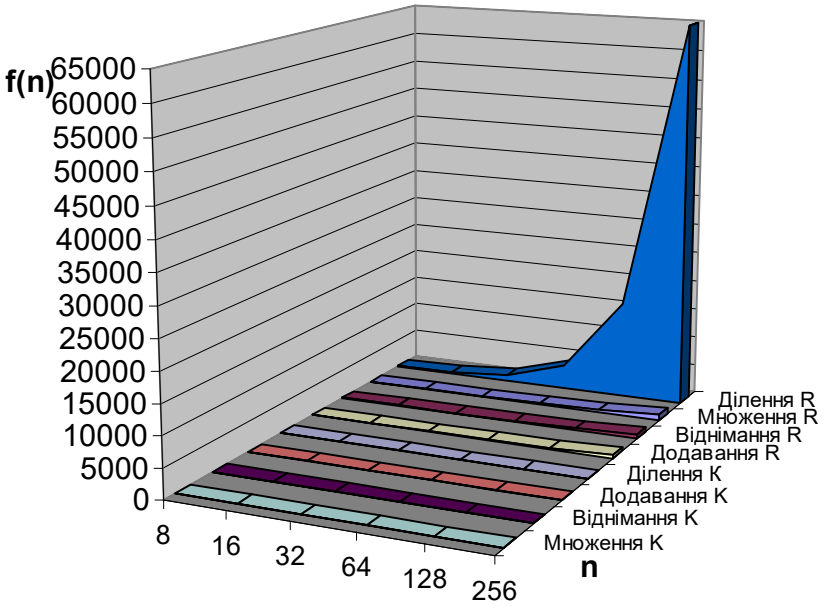


Рисунок 2.2 - Порівняльні характеристики швидкодії виконання арифметико-логічних операцій у базисах Радемахера та цілочисельного перетворення Крестенсона

Недоліком цілочисельної форми перетворення СЗК є практична відсутність простої операції порівняння чисел, що суттєво ускладнює реалізації алгоритмів та відповідних процесів ділення. В той же час, переваги однотактного матричного виконання інших арифметичних операцій забезпечує широкі перспективи застосування теоретичних основ цілочисельного перетворення СЗК для створення та широкомасштабного впровадження супершвидкісних процесорів в комп’ютерних мережах. Особливу перспективу має застосування цілочисельної форми СЗК при створенні спецпроцесорів, в яких базовими операціями є додавання та множення. Прикладом таких процесорів є цифрові фільтри, обчислювачі кореляційних та спектральних характеристик випадкових процесів, а також операції над матрицями та алгебраїчними поліномами.

2.2 Дослідження перетворень та арифметики нормалізованої СЗК

Теоретичною основою утворення нормалізованої форми СЗК (НСЗК) є нормалізація по модулю P обох частин рівняння зворотнього перетворення цілочисельної форми СЗК (2.5):

$$\frac{N_k}{P} = res\sum_{i=1}^{k}\frac{b_i \cdot B_i(\mathrm{mod}\,P)}{P},$$

звідки $[N_k]_0 = res\sum_{i=1}^{k} b_i \cdot \frac{B_i}{P}(\mathrm{mod}1)$, де $0 \le [N_k]_0 \le P-1$; $\frac{B_i}{P} = \frac{1}{p_i}$,

а з врахуванням виразу (2.2), отримаємо:

$$[N_k]_0 = res\sum_{i=1}^{k} b_i \cdot \frac{m_i}{p_i}(\mathrm{mod}1) \quad \text{або} \quad [N_k]_0 = res\sum_{i=1}^{k}[b_i]_0 \cdot m_i(\mathrm{mod}\,1),$$

(2.13)

де $[b_i]_0 = \frac{b_i}{p_i}$, а $0 \le [b_i]_0 < 1$.

(2.14)

Для забезпечення однозначного кодування даних в НСЗК необхідно виконувати умову:

$$\delta_p \le \frac{1}{P}. \tag{2.15}$$

Дана формула визначає необхідне число розрядів після коми, у відповідній системі числення при представленні величини 1/P в нормалізованій формі, тобто

$$\frac{1}{p_i} = 0.\overbrace{gggg}^{n_i}\overbrace{gggg}^{\delta_p}, \tag{2.16}$$

де g – цифри у відповідній системі числення;

n_i – число розрядів до яких заокруглюється результат ділення з приведенням до меншого цілого;

δ_p – дробова частина, яка визначає величину похибки δ_p, якою нехтують.

Таким чином, аналітичний вираз з НСЗК в СЗК отримує вигляд:

$$N_k = \mathrm{int}[N_k]_0 \cdot P, \tag{2.17}$$

де int – символ операції виділення цілої частини.

Кодування даних та виконання арифметичних операцій в НСЗК продемонструємо на прикладі набору модулів:

p_1=3, p_2=5; p_2=7; p_2=8; p_2=11; p_2=5; P=13;$\delta_p \le 0{,}01$.

В таблиці 2.2 показані нормалізовані значення залишків $[b_i]_0$ на основі яких формуються нормалізовані значення чисел $[N_k]_0$ шляхом додавання по mod1.

Таблиця 2.2 – Нормалізовані значення залишків

b_i	p_1=3	p_2=5	p_3=7	p_4=8	p_5=11	p_6=13
0	0.0	0.0	0.0	0.0	0.0	0.0
1	0.1010101010101010101	0.1100110011001100	0.1011011011011011011	0.111	0.10100010111010001010	0.01001110110001001110
2	0.0101010101010101010	0.1001100110011001	0.0110110110110110110	0.110	0.01000101110100010111	0.10011101100010011100
3		0.0110011001100110	0.0010010010010010010	0.101	0.11101000101110100010	0.11101100010011101011
4		0.0011001100110011	0.1101101101101101101	0.100	0.10001011010001011110	0.00111011000100111010
5			0.1001001001001001001	0.011	0.00101110100010111010	0.10001001110110001001
6			0.0100100100100100100	0.010	0.11010001011010001010	0.11011000100111010111
7				0.001	0.01110100010111010001	0.00100111011000100111
8					0.00010111010001011101	0.01110110001001110101
9					0.10111010001011101000	0.11000100111011000100
10					0.01010001011101000101	0.00010011101100010011
11						0.01100010011111000100
12						0.10110001001110110000

Алгоритм виконання операції додавання в НСЗК виконуються на основі наступного математичного виразу при $[x_k]_0+[y_k]_0<1$:

$$[x_k]_0 = \frac{([a_1]_0,[a_2]_0,...,[a_i]_0,...,[a_k]_0)}{+([b_1]_0,[b_2]_0,...,[b_i]_0,...,[b_k]_0)}, \quad (2.18)$$

де $[c_i]_0 = res([a_i]_0 + [b_i]_0) \bmod 1$.

Перевагою НСЗК є виконання операцій над залишками в нормалізованій формі, що спрощує реалізацію процесорів на основі даного базису, за рахунок виключення нелінійних операцій отримання залишку по кожному з модулів

процесора, а також заміни операції по “*mod P*” на операцію по “*mod1*”, яка виконується шляхом простого відкидання цілої частини результату згідно операції int.

2.3 Дослідження числових перетворень Радемахера-Крестенсона у досконалій формі СЗК

Формула перетворення СЗК (2.5) може бути представлена в наступному вигляді:

$$N_k = res\sum_{i=1}^{k} b_i \cdot \frac{P}{p_i} \cdot m_i (\mathrm{mod}\, P), \qquad (2.20)$$

де $0 \le m_i \le p_i - 1$.

Очевидно, що наявність коефіцієнтів m_i в формулі (2.20) ускладнює реалізацію алгоритму виконання цілочисельного перетворення СЗК. Дослідження різних наборів p_i, яким відповідають набори коефіцієнтів m_i, в теоретико-числовому аспекті показали, що існують такі набори модулів p_1, p_2, ... , p_k, які відповідають умовам взаємної простоти з одиничними коефіцієнтами $m_i(m_1=m_2=...=m_i=...=m_k=...=1)$ [114].

Прикладом такого набору модулів є $p_1=2$, $p_2=3$, $p_3=5$, для якого $P=30$, $B_1=15$, $B_2=10$, $B_3=6$, а $m_1=m_2=m_3=1$. В табл.2.3 показаний приклад представлення чисел в досконалій СЗК (ДСЗК) по вказаному наборі модулів:

Таблиця 2.3 – Представлення залишків в досконалій цілочисельній СЗК

N_k	$p_1=2$	$p_2=3$	$p_3=5$
0	0	0	0
1	1	1	1
2	0	2	2
3	1	0	3
4	0	1	4
5	1	2	0
6	0	0	1
…	…	…	…
20	0	2	0
…	…	…	…
28	0	1	3
29	1	2	4

Наприклад, для числа, представленого в ДСЗК, N_k=(1, 2, 4) згідно виразу (2.20) маємо

$$N_k = res(1 \cdot 15 \cdot 1 + 2 \cdot 20 \cdot 1 + 4 \cdot 6 \cdot 1) \bmod 30 = 29.$$

Теоретичною основою досконалої нормалізованої форми СЗК є рівняння (2.20), підставивши в яке $m_1=m_2=m_k=1$, отримаємо базове рівняння перетворення НДСЗК у вигляді:

$$[N_k]_0 = res \sum_{i=1}^{k} [b_i]_0 (\bmod 1). \qquad (2.21)$$

З рівняння (2.21) видно, що в перетворенні НДСЗК виключена операція множення на m_i, а саме перетворення виконується у вигляді сумування нормалізованих залишків $[b_i]_0$ по mod1, що відповідає операції int відкидання цілої частини результату.

Розглянемо приклад застосування НДСЗК для набору модулів: $p_1=2$, $p_2=3$, $p_3=5$ (табл.2.4).

Таблиця 2.4 - Перетворення векторів багатомірного дискретного простору залишків у одномірний дискретний простір Радемахера

N_k	$[b_1]_0$	$[b_2]_0$	$[b_3]_0$	$[N_k]_0$
0	0	0	0	0
1	0.5	0.333	0.2	0.033
2	0	0.666	0.4	0.066
3	0.5	0	0.6	0.1
4	0	0.333	0.8	0.133
5	0.5	0.666	0	0.166
6	0	0	0.2	0.2
7	0.5	0.333	0.4	0.233
…	…	…	…	…
29	0.5	0.666	0.8	0.966

На рис.2.3 показаний приклад визначення $[N_k]_0$ для двох чисел N_{k1}=5, N_{k2}=7.

Операція міжбазисного перетворення може бути представлена у вигляді графа сумування нормалізованих значень залишків в заданій системі модулів. Наприклад: для двох чисел, згідно таблиці 2.4, отримаємо їх коди у

нормалізованій СЗК N_{k1}= (0,5;0,66;0), N_{k2}=(0,5;0,33;0,4) і, згідно графу рис 2.3, отримуємо їх нормалізовані значення у системі модулів (2, 3, 5) $[N_{k1}]_0$= 0,16 і $[N_{k2}]_0$=0,23.

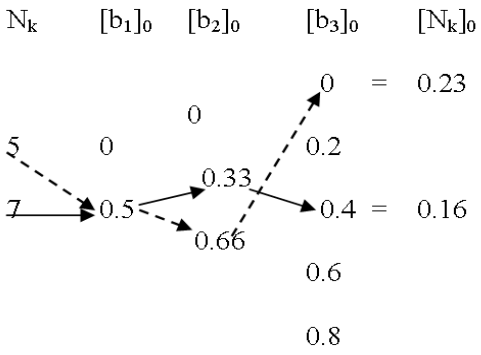


Рисунок 2.3 - Граф визначення $[N_k]_0$

Дані операції виконуються по табличному методу і займають 1 такт. Це дає перевагу перед виконанням тих самих операцій в інших базисах, що дозволяє суттєво спростити спецпроцесор на основі заданого базису.

Як показано у підрозділі 2.2, в якості базового спецпроцесора перетворення Крестенсона-Радемахера доцільно використати алгоритм досконалої нормалізованої форми СЗК (2.21). На рис.2.4 показана процедура міжбазисного перетворення Крестенсона-Радемахера на основі НСЗК.

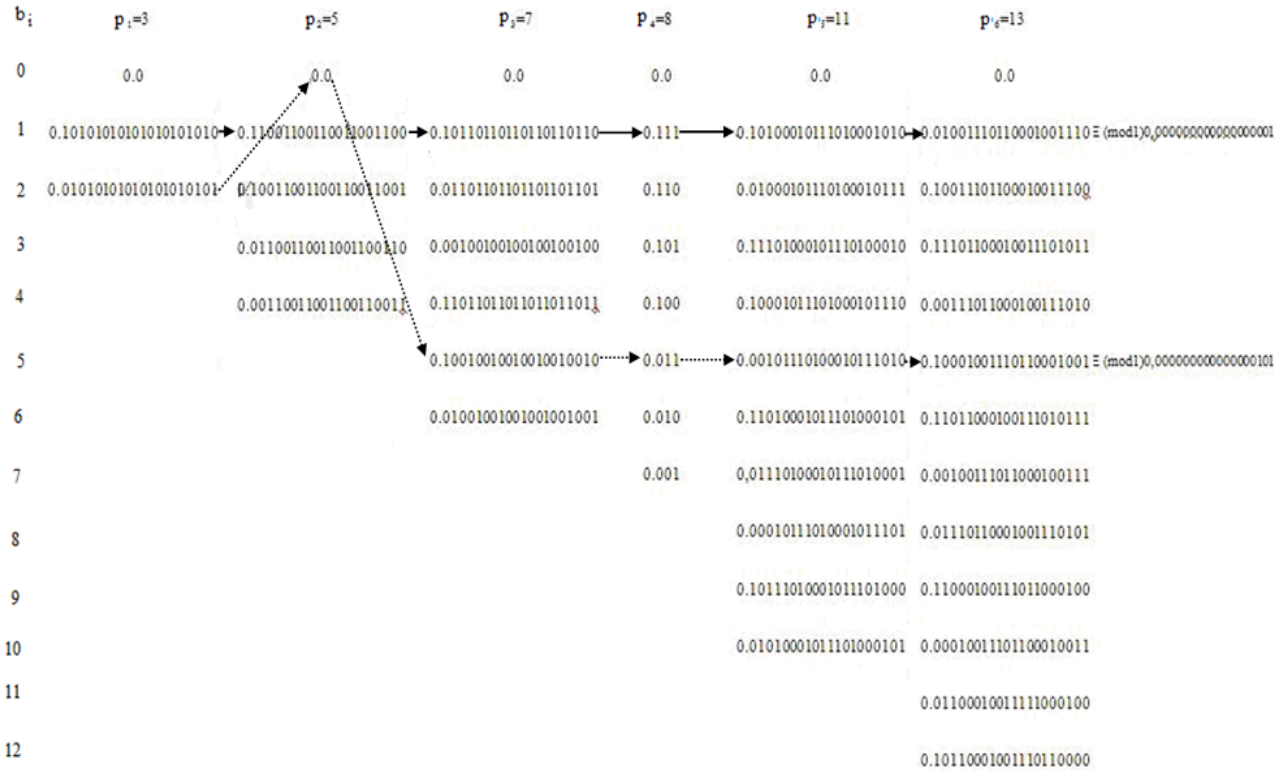


Рисунок 2.4 - Граф обчислення двійкових значень залишків на основі НСЗК

На рис.2.5 приведена діаграма часової складності виконання прямих перетворень досліджених форм СЗК згідно виразів:

$$\tau_k = k \cdot (2\nu(2n+1) + n^2\nu);$$
$$\tau_n = k \cdot (2\nu(2n+1) + 2n\nu);$$
$$\tau_d = k \cdot 2n\nu.$$

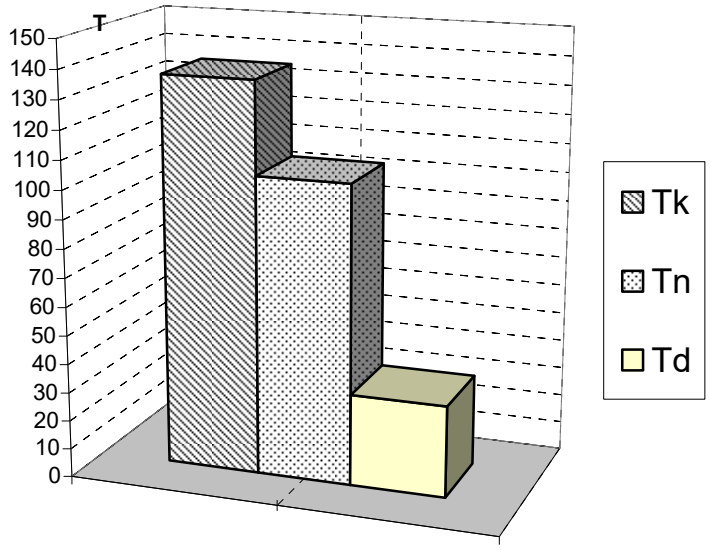


Рисунок 2.5 - Діаграма часової складності виконання перетворень цілочисельної, нормалізованої та досконалої форм СЗК

Проведені дослідження аналітики та часової складності прямих перетворень різних форм СЗК показує, що часова складність НДСЗК на 2 порядки менша по відношенню до цілочисельної форми СЗК, що визначає перспективу застосування НДСЗК для побудови високопродуктивних спецпроцесорів міжбазисного перетворення Крестенсона-Радемахера.

2.4 Розрахунок системи взаємопростих модулів для побудови спецпроцесорів базису Крестенсона різної розрядності

Досвід розробки спецпроцесорів базису Крестенсона на основі модульної цілочисельної арифметики СЗК, які використовувались у системах протиповітряної оборони колишнього Радянського Союзу, як показано в роботі [110], були вибрані наступні набори взаємопростих модулів: 2; 5; 23; 63; 17; 19;

29; 13; 31; 61. При цьому була досягнута швидкодія виконання арифметичних операцій 2,4 млрд/с.

Оскільки спецпроцесори в СЗК можуть мати розрядність 2^k, а їх застосування в задачах шифрування потребує більше 2^{10} розрядів, то науковою задачею є обґрунтування та рекомендація конкретних наборів взаємопростих модулів для спецпроцесорів даного класу. Однією з важливих умов, з точки зору уніфікації схемотехніки таких процесорів, є використання модулів з однаковою розрядністю, що потребує застосування математичних основ теорії чисел [79]. При цьому, при розрядності процесора в базисі Радемахера 2^k, необхідна розрядність спецпроцесора у базисі Крестенсона повинна розраховуватись згідно виразу:

$$N = \hat{E}\left[\log_2(P-1)\right] \geq 2^k + 2,$$

де $\hat{E}[\bullet]$ - цілочисельна функція з заокругленням до більшого цілого;

$P = \prod_{i=1}^{n} p_i$; $p_i \in (p_1, p_2, ..., p_i, ..., p_n)$ - набір взаємопростих модулів з розрядністю $\hat{E}\left[\log_2(P-1)\right] \leq \frac{N}{n}$;

$P \in \overline{0, N-1}$ - діапазон кодування чисел у базисі Крестенсона.

В основу методу вибору системи взаємопростих модулів для великорозрядних процесорів базису Крестенсона покладений наступний алгоритм:

1) вибирається модуль $p_1 = 2^k$ з виконанням умови $\hat{E}\left[\log_2(2^k-1)\right] = n$, оскільки $b_{i\_\max} = p_i - 1$;
2) вибираються всі прості числа розрядністю n в діапазоні $\hat{E}\left[\log_2(2^k-1)\right] \div \hat{E}\left[\log_2(2^{k-1}+1)\right]$, тобто 521, 523, 541, 547, 557, 563, 569, 571, 577, 587, 593, 599, 601, 607, 613, 617, 619, 631, 641, 643, 647, 653, 659, 661, 673, 677, 683, 691, 701, 709, 719, 727, 733, 739, 743, 751, 757, 761, 769, 773, 787, 797, 809, 811, 821, 823, 827, 829, 839, 853, 857, 859, 863, 877, 881, 883,

887, 907, 911, 919, 929, 937, 941, 947, 953, 967, 971, 977, 983, 991, 997,1009, 1013, 1019, 1021 при *n=10*;

3) вибираються добутки простих чисел, сумарна розрядність яких відповідає n-біт;
4) вибираються прості числа розрядність яких дорівнює (n-1)-біт.

В результаті використання розробленого алгоритму будуть отримані набори модулів для процесорів різної розрядності (додаток В), приклад наборів модулів для процесорів з розрядністю 256, 512 та 1024 біти представлені в табл.2.5.

Таблиця 2.5 – Набори модулів для 256, 512 та 1024-бітного спецпроцесора

Розрядність процесора	Набори модулів
256 біт	8 бітні: 256, 255, 253, 251, 247, 241, 239, 233, 229, 227, 223, 217, 211, 199, 197, 193, 191, 181, 179, 173, 167, 163, 157, 151, 149, 139, 137, 131, 127, 113, 109, 107,103, 101, 97. 9 бітні: 1) 512, 511, 509, 505, 503, 501, 499, 493, 491, 487, 481, 479, 473, 467, 463, 461, 457, 449, 443, 439, 437, 433, 431, 421, 419, 409, 401, 397, 389. 2) 263, 269, 271, 277, 281, 283, 293, 307, 311, 313, 317, 331, 337, 347, 349, 353, 359, 367, 373, 379, 383, 389, 397, 401, 409, 419, 421, 431, 433, 437, 439.
512 біт	9 бітні: 512, 511, 509, 505, 503, 501, 499, 493, 491, 487, 481, 479, 473, 467, 463, 461, 457, 449, 443, 439, 437, 433, 431, 421, 419, 409, 401, 397, 389, 383, 379, 373, 367, 359, 353, 349, 347, 337, 331, 317, 313, 311, 307, 293, 283, 281, 277, 271, 269, 263, 257, 251, 241, 239, 233, 229, 227, 223, 211, 199, 197.
1024 біти	10 бітні: 1024, 1023, 1021, 1019, 1013, 1009, 1007, 1003, 997, 995, 991, 989, 983, 977, 973, 971, 967, 953, 949, 947, 941, 937, 929, 919, 911, 907, 887, 883, 881, 877, 863, 859, 857, 853, 841, 839, 829, 827, 823, 821, 811, 809, 797, 787, 773, 769, 761, 757, 751, 743, 739, 733, 727, 719, 709, 701, 691, 683, 677, 673, 661, 659, 653, 647, 643, 641, 631, 619, 617, 613, 607, 601, 599, 593, 587, 577, 571, 569, 563, 557, 547, 541, 523, 521, 509, 503, 499, 479, 467, 463, 461, 457, 449, 443, 439, 433, 431, 421, 419, 409, 401, 397, 389, 383, 379, 373, 367.

2.5 Розробка теоретичних основ розмежування розрядної сітки процесора у базисі Радемахера-Крестенсона

Розмежовану систему можна ефективно застосовувати для побудови спецпроцесорів, які включають операції сумування, множення та обчислення залишків по модулю.

Перспективним є створення класу високопродуктивних спецпроцесорів для опрацювання даних великої розрядності (256 – 10^9біт). Типовими задачами, які реалізують такий клас спецпроцесорів, є процесори шифрування та дешифрування інформаційних потоків у системах захисту інформації, розв'язання систем великої розмірності, задачі знаходження найбільших дільників двох великорозрядних чисел, пошуку простих чисел великої розрядності та ряд інших аналогічних задач.

Теоретичною основою РСЗК є цілочисельна форма СЗК, рівняння якої представлено у вигляді суми [114]:

$$N_k = N_{1k} + N_{2k} + ... + N_{ik} + ... + N_{nk},$$

де N_{ik} – m- розрядний (розмежований) фрагмент числа N_k, яке представлене у двійковій системі числення, числового базису Радемахера.

Наприклад, 1024-х розрядний процесор СЗК може бути розмежований на 32 фрагменти по 32 біти (рис.2.6).

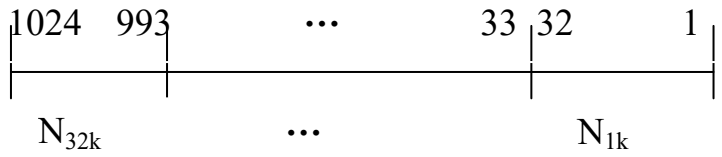
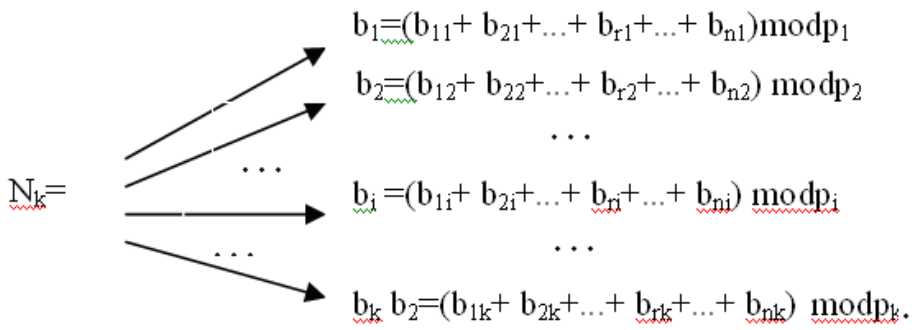


Рисунок 2.6 - Приклад розмежування 1024-х розрядного процесора

Таким чином, пряме перетворення РСЗК отримає вигляд :



При цьому математичні операції над числами в РСЗК можуть бути розмежовані по кожному із фрагментів процесора, що забезпечує ще більш глибокий рівень розпаралелювання обробки інформації, а відповідно підвищення швидкодії процесора СЗК.

Зі структури розмежованого процесора зрозуміло, що вона потребує обчислення залишків для кожного компонента згідно виразу

$$b_{ij} = resN_{ij} (\text{mod } p_i).$$

Звідки, процедура обчислення загального залишку виконується згідно виразу

$$b_i = res(b_{i1} + b_{i2} + ... + b_{in}) \text{mod } p_i.$$

При бінарному розмежуванні двійкових чисел базису Радемахера, тобто k=0, структура розмежування має наступний вигляд:

32	•••	i	•••	3	2	1
N32		Nik		N3	N2	N1

.

Запропонований в [89] алгоритм перетворення чисел базису Радемахера в СЗК на основі теорії РСЗК дозволяє поглибити процес розпаралелення та спрощення арифметичних операцій базису Крестенсона. В той же час, виконання процедури розмежування на N розрядів

$$N = 2^i \cdot n,$$

де N – число розрядів процесора, 2^i – коефіцієнт розмежування, приводить, як це показано в [67, 117], до спрощення складних дешифраторів, а також передбачає наскрізні переноси в РСЗК.

Реалізація побітного розмежування чисел в базисі Радемахера дозволяє суттєво спростити алгоритм переходу з базису Радемахера в базис Крестенсона,

а також реалізувати повнофункціональну арифметичну операцію у РСЗК без наскрізних переносів та при максимальному розпаралеленні.

Наявність значних переваг в СЗК, які приводять до створення модульних дешифраторів системи залишкових класів, які викладені в [37], визначає ефективність використання розмежованої СЗК, що дозволяє поглибити процес розпаралелення, спрощення виконання арифметичних операцій та реалізації дешифраторів на основі СЗК.

Основними операціями, які повинен виконувати спецпроцесор, незалежно від його застосування, є арифметичні операції додавання, множення, віднімання та ділення. Проте, для розробки алгоритмів виконання операцій віднімання та множення необхідна проста логічна операція порівняння. Отже, основними на даному етапі проектування спецпроцесора є реалізація операцій сумування та порівняння.

В загальному випадку для двох чисел $x = x_{n-1}...x_i...x_0$ та $y = y_{n-1}...y_i...y_0$ алгоритм здійснення операції сумування у розмежованій СЗК буде виконуватися наступним чином [68]:

1. Для кожного з бітів знаходимо залишки по модулях (2, 3, 5, 7,…, P_j,…, P_k). В результаті чого формуються матриці залишків числа X (табл.2.7) і залишків числа Y (табл.2.8).

Таблиця 2.7 - Матриця залишків числа X

	X_{n-1}	X_{n-2}	…	X_i	…	X_1	X_0
P_1	$a_{n-1,1}$	$a_{n-2,1}$	…	$a_{i,1}$	…	$a_{1,1}$	$a_{0,1}$
P_2	$a_{n-1,2}$	$a_{n-2,2}$	…	$a_{i,2}$	…	$a_{1,2}$	$a_{0,2}$
P_3	$a_{n-1,3}$	$a_{n-2,3}$	…	$a_{i,3}$	…	$a_{1,3}$	$a_{0,3}$
P_4	$a_{n-1,4}$	$a_{n-2,4}$	…	$a_{i,4}$	…	$a_{1,4}$	$a_{0,4}$
…	…	…	…	…	…	…	…
P_j	$a_{n-1,j}$	$a_{n-2,j}$	…	$a_{i,j}$	…	$a_{1,j}$	$a_{0,j}$
…	…	…	…	…	…	…	…
P_k	$a_{n-1,k}$	$a_{n-2,k}$	…	$a_{i,k}$	…	$a_{1,k}$	$a_{0,k}$

У табл.2.7 зображено структуру відображення числа X, де X_j – біти досліджуваного числа X представленого у вигляді залишків a_j по модулях P_j, відповідних бітів. Тобто, число розбивається на послідовність бітів і подальші

операції виконуються з кожним бітом окремо, отримуючи доступ до нього через занесену в пам’ять спецпроцесора матрицю.

Таблиця 2.8 - Матриця залишків числа Y

	Y_{n-1}	Y_{n-2}	…	Y_i	…	Y_1	Y_0
P_1	$b_{n-1,1}$	$b_{n-2,1}$	…	$b_{i,1}$	…	$b_{1,1}$	$b_{0,1}$
P_2	$b_{n-1,2}$	$b_{n-2,2}$	…	$b_{i,2}$	…	$b_{1,2}$	$b_{0,2}$
P_3	$b_{n-1,3}$	$b_{n-2,3}$	…	$b_{i,3}$	…	$b_{1,3}$	$b_{0,3}$
P_4	$b_{n-1,4}$	$b_{n-2,4}$	…	$b_{i,4}$	…	$b_{1,4}$	$b_{0,4}$
…	…	…	…	…	…	…	…
P_j	$b_{n-1,j}$	$b_{n-2,j}$	…	$b_{i,j}$	…	$b_{1,j}$	$b_{0,j}$
…	…	…	…	…	…	…	…
P_k	$b_{n-1,k}$	$b_{n-2,k}$	…	$b_{i,k}$	…	$b_{1,k}$	$b_{0,k}$

У табл.2.8 зображено структуру відображення числа Y, де Y_j – біти досліджуваного числа Y представленого у вигляді залишків b_j по модулях P_j, відповідних бітів. Таке представлення дозволяє використати дану систему для створення алгоритмів перевірки чисел виду $N_i = 2^n + 1$, коли $n \in \overline{0,\infty}$. Даний тип чисел належить до чисел класу Ферма виду $1000...0001_{(2)}$.

Визначення 1. Парні числа при $2^0 = 0$, тобто в першому розряді біт числа $N_1 = 0$, $N_i = 2 \cdot j$, де $j = 1,2,...\infty$. Тобто 50% всіх чисел $0 \to \infty$ не належить до простих.

Визначення 2. Двійкові степені бітів чисел N, які відповідають умові $resN(\mathrm{mod}\,3) = 2$, діляться на 3, оскільки $res(2^{2i+1} + 2^0)(\mathrm{mod}\,3) = 0$. В результаті отримаємо таблицю ділимості чисел по модулю 3 (табл.2.9).

Таблиця 2.9 - Ділимість чисел на 3

	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…	…						…					…				
k+16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…	…						…					…				

Для чисел, які відповідають умові $res2^i(\mathrm{mod}\,5)=4$, що діляться на 5, оскільки $res(2^{4i+2}+2^0)(\mathrm{mod}\,5)=0$, формуємо таблицю 2.10.

Таблиця 2.10 - Ділимість чисел на 5

	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…		…					…					…				
k+16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…		…					…					…				

Визначення 3. Даний тип чисел ніколи не ділиться на 7 без залишку, оскільки в РСЗК описується наступним рядом чисел:

…	1	4	2	1	4	2	1

,

тобто відсутній залишок рівний 6.

В результаті з отриманих 50% віднімаємо 25% і отримуємо четверту частину всіх чисел (табл.2.11), які описуються аналітичним рівнянням $n=2+k*4$, де $k=0,1,2...\infty$.

Таблиця 2.11 - Ділимість чисел на 3 та 5

	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…		…					…					…				
k+16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…		…					…					…				

Для дослідження чисел ділимості на 11 розглянемо залишки РСЗК по відповідному модулю, що описуються наступним рядом чисел:

…	10	5	8	4	2	1	6	3	7	9	10	5	8	4	2	1

.

Тобто всі числа, які в степені РСЗК $res2^i(\mathrm{mod}11)=10$, діляться на 11, оскільки $res(2^{10i+1}+2^0)(\mathrm{mod}11)=0$. Отже, отримаємо таблицю ділимості чисел на 11 (табл.2.12).

Таблиця 2.12 - Ділимість чисел на 11.

	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
64	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
80	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
96	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…	…						…				…					

З табл. 2.13 видно, що числа, які діляться на 11, потрапляють в діапазон бітів залишків по модулях 3 та 5 з степенями бітів 1+k16, 4+k16, 8+k16.

Табл.2.13 – Ділимість чисел на 3, 5 та 11.

	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}
16	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
48	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
64	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
80	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
96	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
…	…						…				…					

Повторивши даний алгоритм для наступних простих чисел отримаємо таблиці простих чисел даного класу, які можуть бути використані для формування системи взаємопростих модулів СЗК базису Крестенсона.

ВИСНОВКИ ДО РОЗДІЛУ 2

1. Викладені теоретичні засади прямого та зворотного перетворення цілочисельної системи залишкових класів базису Крестенсона. Поданий приклад розрахунку базисних чисел B_i та нормуючі коефіцієнтів m_i для взаємопростих модулів p_1, p_2. Формалізована аналітика арифметичних операцій додавання, віднімання, множення в СЗК та розраховані порівняльні характеристики швидкодії виконання арифметико-логічних операцій у базисах Радемахера та Крестенсона, в результаті чого встановлено, що операції сумування та множення в базисі Крестенсона можуть виконуватись за один такт роботи процесора, а при зростанні розрядності опрацьовуваних чисел більше 128 біт швидкодія процесорів базису Крестенсона перевищує відповідні характеристики процесорів базису Радемахера на 2-3 порядки.

2. Досліджено пряме та зворотне перетворення нормалізованої форми СЗК та визначені умови необхідної розрядності представлення двійкових кодів нормалізованих залишків $[b_i]_0$ двійковими кодами з фіксованою комою. Розраховано приклад кодування чисел в нормалізованій СЗК та формалізований алгоритм виконання операції додавання в НСЗК. Показано, що перевагою нормалізованої СЗК є виключення операції з великою часовою та апаратною складністю по базовому модулю СЗК і заміна її операцією по *mod*1, яка виконується шляхом відкидання цілої частини нормалізованих залишків.

3. В результаті дослідження досконалої форми СЗК встановлено та показано на діаграмі, що дана форма СЗК забезпечує зменшення часової складності прямих перетворень в 5 та 10 разів у порівнянні з нормалізованою та цілочисельною формами СЗК. Таким чином обґрунтована доцільність та ефективність застосування досконалих форм СЗК для побудови спецпроцесорів у базисі Крестенсона, а також процесорів міжбазисних перетворень Крестенсона-Радемахера.

4. Розроблено алгоритм операції ділення на основі бінарно-розмежованої СЗК та отримана аналітична матриця цієї операції у базисі Крестенсона, що дозволило замінити операцію ділення операцією множення у

розширеній СЗК та доповнюючими кодами залишків і виконувати її обмежене число тактів процесора з підвищенням швидкодії виконання операції ділення на 2-3 порядки.

5. Обґрунтована перспектива розмежування великорозрядних чисел з метою зменшення апаратної та часової складності процесорів міжбазисних перетворень та опрацювання великорозрядних чисел.

РОЗДІЛ 3

РОЗРОБКА ТА ДОСЛІДЖЕННЯ МІЖБАЗИСНИХ ПЕРЕТВОРЕНЬ ВЕЛИКОРОЗРЯДНИХ ЧИСЕЛ У РОЗМЕЖОВАНІЙ СЗК

3.1 Дослідження часової складності міжбазисних перетворень ТЧБ, які породжують системи числення

В роботах [37, 122] проведені дослідження теорії міжбазисних перетворень у різних ТЧБ, які породжують системи числення. В табл.3.1 приведені отримані аналітичні вирази міжбазисних перетворень у різних ТЧБ.

Таблиця 3.1 - Міжбазисні перетворення різних ТЧБ

№	Теоретико-числові базиси	Алгоритми міжбазисних перетворень
1	2	3
1	Радемахера-Крестенсона (цілочисельна)	$N_k=(a_{n-1},...,a_i,..,a_0); a_i\in\overline{0,1}; N_k=\sum_{i=0}^{n-1}a_i\cdot 2^i;$ $N_k=\rightarrow b_i$ $b_i=resN_k(\mathrm{mod} p_i);$ $N_k=a_ip_i+b_i,\ P=\prod_{i=1}^{k}p_i;\ 0\le N_k\le P,\ p_i\equiv p_j.$
2	Радемахера-Крестенсона (нормалізована)	$\frac{N_k}{P}=res\sum_{i=1}^{k}\frac{b_i\cdot B_i(\mathrm{mod}P)}{P},\ [N_k]_0=res\sum_{i=1}^{k}b_i\cdot\frac{B_i}{P}(\mathrm{mod}\,1),$ $0\le[N_k]_0\le P-1;\ \frac{B_i}{P}=\frac{1}{p_i},$ $\delta_p\le\frac{1}{P},\frac{1}{p_i}=0\overbrace{ggggggg}^{n}\overbrace{g}^{\delta}.$
3	Радемахера-Крестенсона (досконала)	$[N_k]_0=res\sum_{i=1}^{k}[b_i]_0(\mathrm{mod}1).$ $b_i=\mathrm{int}\,res[N_k]_0(\mathrm{mod}\,1)\cdot P_i$
4	Радемахера-Крестенсона (розмежована)	$N_k=N_{1k}+N_{2k}+...+N_{ik}+...+N_{nk},$ $b_{ij}=resN_{ij}(\mathrm{mod}\,p_i), P=\prod_{i=1}^{k}p_i$, де $0\le N_k\le P-1$
5	Крестенсона-Радемахера (цілочисельна)	$N_k=res\sum_{i=1}^{k}b_i\cdot B_i(\mathrm{mod}\ P), B_i=\frac{P}{p_i}\cdot m_i\equiv 1(\mathrm{mod}\ \ p_i)\cdot$
6	Крестенсона-Радемахера (нормалізована)	$[N_k]_0=res\sum_{i=1}^{k}b_i\cdot\frac{m_i}{p_i}(\mathrm{mod}\,1),\ [N_k]_0=res\sum_{i=1}^{k}[b_i]_0\cdot m_i(\mathrm{mod}\,1),$ $[b_i]_0=\frac{b_i}{p_i},\ 0\le[b_i]_0\le 1,\ N_k=\mathrm{int}[N_k]_0\cdot P,$

Продовження таблиці 3.1

1	2	3
7	Крестенсона-Радемахера (досконала)	$[N_k]_0 = res\sum_{i=1}^{k}[b_i]_0 (\mathrm{mod}\, 1).$
8	Крестенсона-Радемахера (розмежована)	$N_k = res\sum_{r=1}^{n}\sum_{i=1}^{k} res(b_{1i} + b_{2i} + \ldots + b_{ri} + \ldots + b_{ni}) \bmod P_i \cdot B_i \bmod P.$
9	Радемахера-Галуа	$N_k \to \sum_{i=0}^{n-1} i \to G_0, G_1 \ldots G_{i-1}; G_0 = (111\ldots1); G_{i+1} = G_i \oplus G_{i-n}; i \in \overline{0, n-1}.$
10	Крестенсона-Галуа	$N_k = (b_1, b_2, \ldots, b_i, \ldots, b_k) \to \sum_{i=0}^{n-1} i \to G_i, G_{i-1} \ldots G_{i-n};$ $G_0 = (111\ldots1); G_{i+1} = G_i \oplus G_{i-n}; i \in \overline{0, n-1}.$
11	Галуа-Радемахера	$G_i, G_{i-1} \ldots G_{i-n} \to \sum_{i=0}^{n-1} i \to N(a_{n-1}, \ldots, a_i, \ldots, a_0) \to N_k; a_i \in \overline{0,1}.$
12	Галуа-Крестенсона	$G_i, G_{i-1} \ldots G_{i-n} \to \sum_{i=0}^{n-1} i \to N_k = (b_1, b_2, \ldots, b_i, \ldots, b_k).$

На рис. 3.1 показана розрахована на основі аналітичних виразів табл.3.1 діаграма часової складності алгоритмів міжбазисних перетворень досліджуваних ТЧБ.

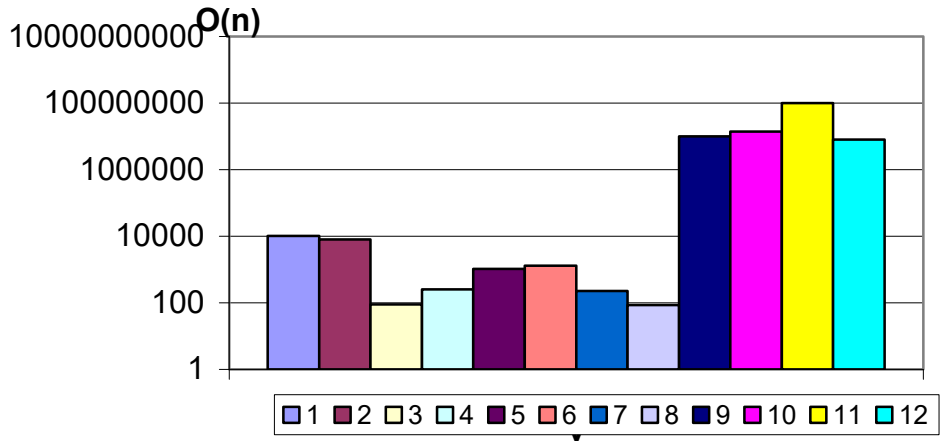


Рисунок 3.1 - Діаграма часової складності міжбазисних перетворень різних ТЧБ

В результаті проведених досліджень встановлено, що перетворення Радемахера-Крестенсона та Крестенсона-Радемахера характеризуються

найменшою часовою складністю та найбільшим числом міжбазисних перетворень у цілочисельній, нормалізованій, досконалій та розмежованій формах СЗК. Тому такі міжбазисні перетворення можуть бути ефективно застосовані при створенні мультибазисних процесорів, а також опрацюванні великорозрядних чисел в системах криптозахисту інформації та ряді фундаментальних задач теорії чисел.

3.2 Розробка високопродуктивних матричних операцій сумування у розмежованій СЗК

Для виконання операції додавання застосовуємо бінарно-розмежовану СЗК, на основі якої формуємо матриці по кожному біту двійкового числа X (Xn-1, Xn-2, …, Xi,…, X1, X0) та Y (Yn-1, Yn-2, …,Yi,… ,Y1, Y0), звідки формується матриця залишків кожного і-того розряду у системі взаємопростих модулів P1, P2, ... Pj, ... Pk [87].

Виконуємо операцію сумування над матрицями за формулою:

$$(\sum_{i=1}^{n-1} b_{i,j} + \sum_{i=0}^{n-1} a_{i,j}) \bmod P_j = C_{i,j};\ i \in \overline{0},\ n-1;\ j \in \overline{1,k}. \qquad (3.1)$$

При цьому $a_{i,j}$ – залишок біту X_i по модулю P_j, $b_{i,j}$ - залишок біту Y_i по модулю P_j, i – порядковий номер біту, j – порядковий номер модуля, по якому шукається залишок.

Тобто, виконується сумування залишків по відповідних модулях:

$$\begin{array}{rc} & P_1, P_2, \ldots P_i, \ldots P_k \\ x_k & (a_1, a_2, \ldots a_i, \ldots a_k) \\ & + \ + \quad + \quad + \\ y_k & (b_1, b_2, \ldots b_i, \ldots b_k) \\ & (c_1, c_2, \ldots c_i, \ldots c_k) \end{array}.$$

В результаті отримуємо суму двох чисел X та Y у вигляді матриці залишків РСЗК (табл. 3.2).

Таблиця 3.2 - Матриця залишків результату сумування

P_1	$C_{n-1,1}$	$C_{n-2,1}$	…	$C_{i,1}$	…	$C_{1,1}$	$C_{0,1}$
P_2	$C_{n-1,2}$	$C_{n-2,2}$	…	$C_{i,2}$	…	$C_{1,2}$	$C_{0,2}$
P_3	$C_{n-1,3}$	$C_{n-2,3}$	…	$C_{i,3}$	…	$C_{1,3}$	$C_{0,3}$
P_4	$C_{n-1,4}$	$C_{n-2,4}$	…	$C_{i,4}$	…	$C_{1,4}$	$C_{0,4}$
…	…	…	…	…	…	…	…
P_j	$C_{n-1,j}$	$C_{n-2,j}$	…	$C_{i,j}$	…	$C_{1,j}$	$C_{0,j}$
…	…	…	…	…	…	…	…
P_k	$C_{n-1,k}$	$C_{n-2,k}$	…	$C_{i,k}$	…	$C_{1,k}$	$C_{0,k}$

Алгоритм трансформації операції сумування з РСЗК в цілочисельній формі СЗК обчислюється згідно операції:

$$(C_{n-1,j} + C_{n-2,j} + ... + C_{i,j} + ... + C_{1,j} + C_{0,j}) \bmod P_j = \gamma_j, \qquad (3.2)$$

де γ_j - відповідно залишок в цілочисельній формі СЗК.

Розроблено алгоритм виконання матрично-модулярної операції множення чисел з розрядністю n біт $a=a_{n-1}2^{n-1}+\ldots+ +a_i2^i+\ldots+ a_12+a_0$ та $b=b_{n-1}2^{n-1}+\ldots+ +b_i2^j+\ldots+ +b_12+b_0$, де a_i, b_j=0, 1, n–розрядність модуля р. Обчислення результату їх множення за модулем р відбувається на основі матриці $\left\|c_{ij}\right\| = 2^{i+j} \bmod p$ (табл. 3.3). Результат операції множення чисел а та·b отримується згідно формули:

$$a \cdot b = \left(\sum_{i=1}^{n-1} \sum_{j=1}^{n-1} a_i b_j 2^{i+j} \right) \bmod \ \ p,$$

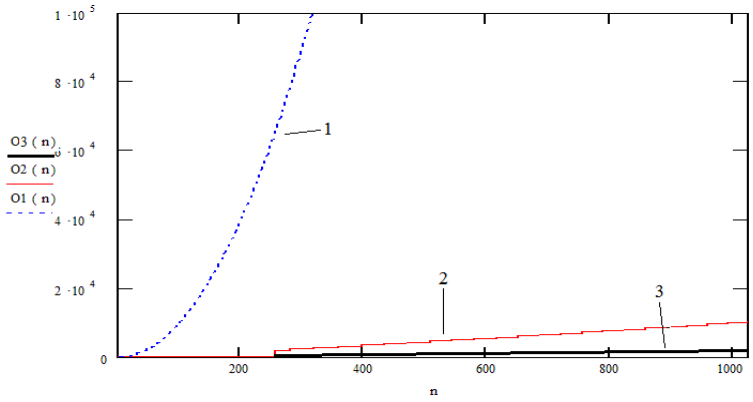
де a_i, $b_j=1$, тобто c_{ij} знаходиться в місці перетину стовпця та рядка, для яких відповідні a_i та b_j дорівнюють 1

Таблиця 3.3 – Матриця множення в базисі Радемахера–Крестенсона

	b_{n-1}	…	b_j	…	b_1	b_0
a_{n-1}	$c_{n-1\ n-1}$	…	$c_{n-1\ j}$	…	$c_{n-1\ 1}$	$c_{n-1\ 0}$
…	…	…	…	…	…	…
a_i	$c_{i\ n-1}$	…	c_{ij}	…	c_{i1}	c_{i0}
…	…	…	…	…	…	…
‘a_1	$c_{1\ n-1}$	…	c_{1j}	…	c_{11}	c_{10}
a_0	$c_{0\ n-1}$	…	c_{0j}	…	c_{01}	c_{00}

Таким чином, отриманий новий алгоритм заміни операції множення, яка має лінійно-логарифмічну обчислювальну складність $O1(n) = n \log n$, операцією

додавання з логарифмічною складністю $O2(n)=\begin{cases}(\log_2 n)^2, \text{якщо\_}n<256\\ n\cdot(\log_2 n), \text{в інших випадках}\end{cases}$ та для спецпроцесора в РСЗК $O3(n)=2\log n$. Результати дослідження ефективності даного алгоритму наведені на рис. 3.2.



1 – універсальний процесор у базисі Радемахера;
2 – універсальний процесор у базисі Крестенсона;
3 – спецпроцесор у розмежованій СЗК базису Крестенсона.

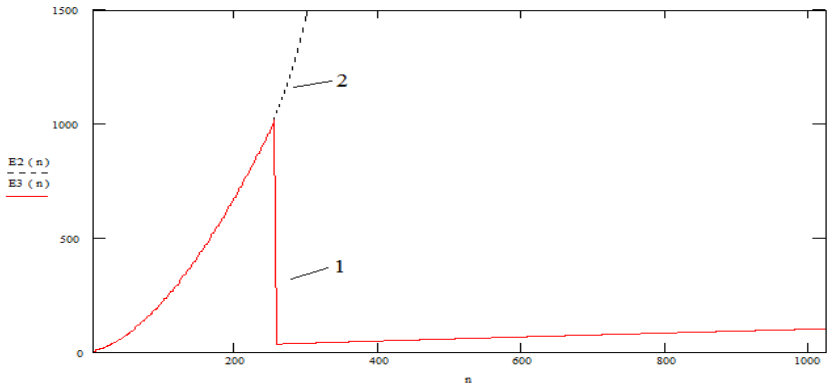
Рисунок 3.2 - Складність операції модулярного множення

З рис.3.2 видно, що при зростанні розмірності параметрів алгоритму модулярного множення відомий алгоритм не може бути фізично реалізований з використанням сучасної мікропроцесорної техніки, особливо з врахуванням наступної операції піднесення до степеня, яка включає множення. А запропонований алгоритм з використанням базису Крестенсона дозволяє, за рахунок нарощення розмірності ключів, параметрів асиметричних крипто алгоритмів, суттєво підвищити рівень захисту інформаційних потоків від несанкціонованого доступу, які ґрунтуються на незворотних перетвореннях дискретного логарифмування або факторизації великорозрядних чисел [7].

Ефективність запропонованого алгоритму модулярного множення буде визначатися як співвідношення обчислювальних складностей

$$E3(n)=\begin{cases}\dfrac{n^2}{(\log_2 n)^2}, \text{якщо } n<256\\ \dfrac{n^2}{n\cdot\log_2 n}, \text{ в інших випадках.}\end{cases}$$

, що показано на рис.3.3.



1 – емуляція для спецпроцесора;
2 – емуляція на універсальному процесорі.

Рисунок 3.3 - Ефективність реалізації алгоритму модулярного множення розмірності n

Проаналізувавши співвідношення складностей та результатів рис.3.3, можна зробити висновок, що в діапазоні розрядності вхідних даних алгоритму модулярного множення від 0 до 256 бітів ефективність стрімко зростає при переході від двійково-десяткової системи числення з квадратичною часовою складністю до розмежованої системи числення Радемахера-Крестенсона з логарифмічною. При зростанні розрядності чисел більших від 256 біт складність операції модулярного множення на основі використання ТЧБ Радемахера-Крестенсона буде лінійно-логарифмічною. Слід зазначити, що в основу розробленого алгоритму входить додавання чисел з великою розрядністю, який на відміну від існуючих замінює операцію множення, яка здійснюється за n^2 тактів, n-тактною операцією додавання.

Очевидно, що при створенні спецпроцесорів шифрування інформації з розрядністю 1024 та 2048 біт втрата швидкодії (див. рисунки 3.2, 3.3), яка спостерігається на універсальних процесорах з меншою розрядністю буде компенсована підвищенням швидкодії арифметико-логічних пристроїв відповідної розрядності.

Отже, використання базису Крестенсона суттєво збільшує перспективи щодо розробки систем з високим рівнем захисту інформаційних потоків і дозволяє зменшити часову та обчислювальну складність виконання алгоритму модулярного множення.

3.3 Вдосконалення та дослідження компонентів спецпроцесорів модульного опрацювання великорозрядних чисел у бінарно-розмежованій СЗК

3.3.1 Матрично-модульний суматор та перемножувач у базисі Хаара

Вимоги створення швидкодіючих спецпроцесорів опрацювання великорозрядних чисел у базисі Крестенсона визначають доцільність застосування однотактних матричних суматорів по модулю в кодах базису Хаара. Алгоритми виконання операцій сумування та множення по модулю у базисі Хаара здійснюються згідно аналітичного виразів:

$S_k(H) = res(b_i + b_j) \bmod P$, $Z_k(H) = res(b_i \cdot b_j) \bmod P$;

$S_k = k(H)$, якщо $(b_i + b_j) = P_j + k$, що відповідає $b_i(H) \wedge b_j(H) = 1$;

$Z_k = k(H)$, якщо $(b_i \cdot b_j) = P_j + k$, що відповідає $b_i(H) \wedge b_j(H) = 1$.

На рис.3.4 та 3.5 показані структури матрично-модульного суматора та перемножувача базису Хаара по модулю 11.

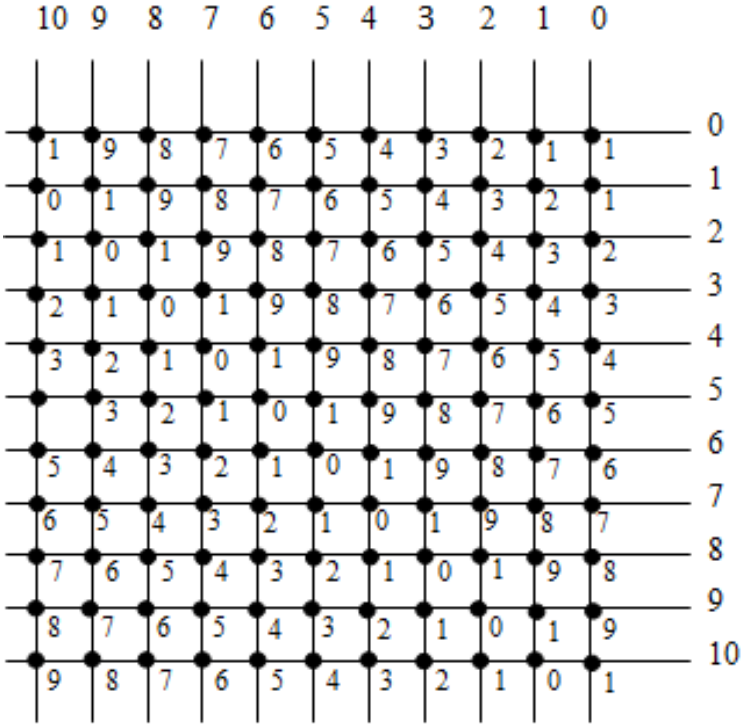


Рисунок 3.4 - Матрично-модульний суматор базису Хаара

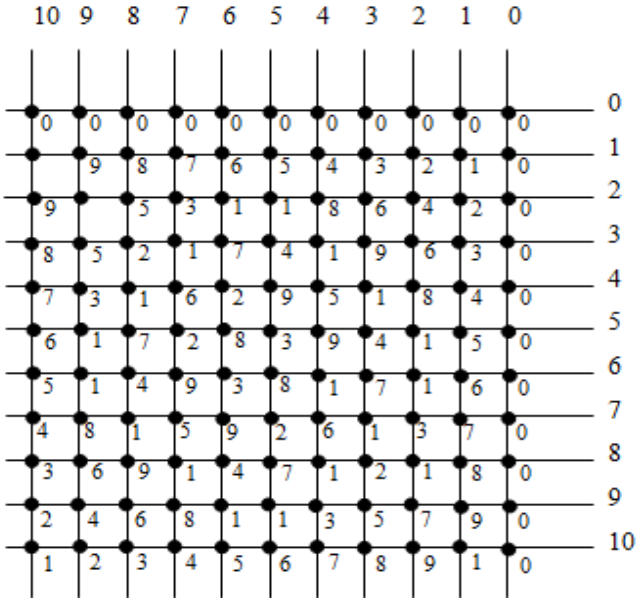


Рисунок 3.5 - Матрично-модульний перемножувач базису Хаара

Оскільки на пересіченні шин матриці (див. рис.3.4, 3.5) розміщається один логічний елемент «І-НЕ», а вихідний біт формується на основі провідного логічного елемента «АБО», то апаратна складність таких пристроїв рівна $P^2 + P$ та P^2 (рис.3.6), а часова складність рівна $2v$.

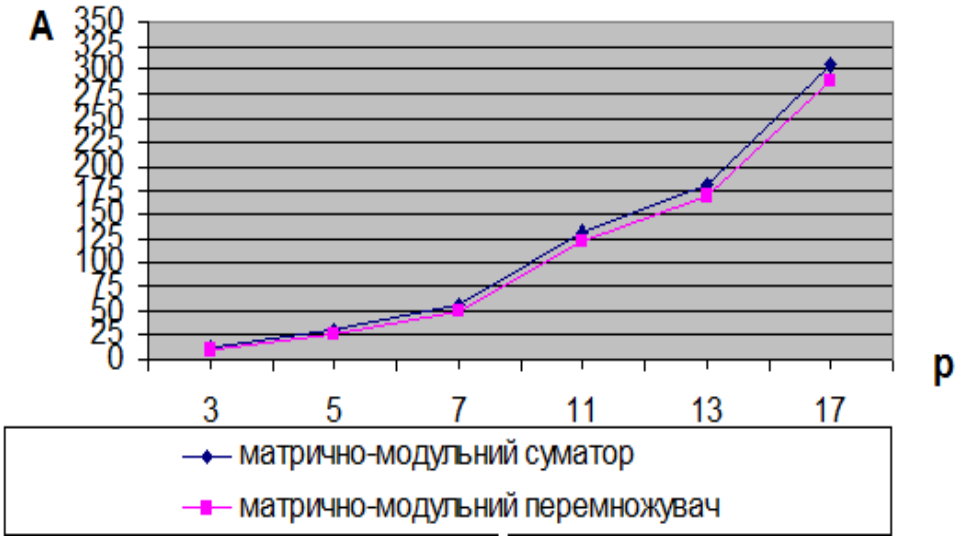


Рисунок 3.6 - Апаратна складність матрично-модульного суматора та перемножувача у базисі Хаара

Швидке зростання апаратної складності даного класу матрично-модульних пристроїв не позволяє використовувати їх автономно при опрацюванні великорозрядних чисел. В той же час, використання таких пристроїв у

процесорах базису Крестенсона є достатньо ефективним, оскільки при переході до базису Крестенсона число вентильних елементів замість N^2 буде рівне сумі P_j^2 , де P_j – взаємопрості модулі, добуток яких перевищує число N. На рис.3.7 показано характеристики часової складності процесорів виконання операцій сумування та множення по модулю у базисах Хаара та Крестенсона в залежності від значення числа N.

При створенні спецпроцесорів опрацювання великорозрядних чисел k≥1024 біт, для яких необхідно використовувати матриці по модулю, коди залишків та взаємнопростих модулів можуть перевищити розрядність 10 біт, що приводить до швидкого зростання складності апаратної реалізації матриць модульних операцій. У даному випадку доцільно виконувати модульні операції над 10-и бітними кодами залишків у базисі Радемахера по кожному модулю системи залишкових класів, що на порядок може зменшити швидкодію процесорів базису Крестенсона. При цьому, апаратна складність по кожному каналу процесора суттєво, практично на 2 порядки, зменшиться.

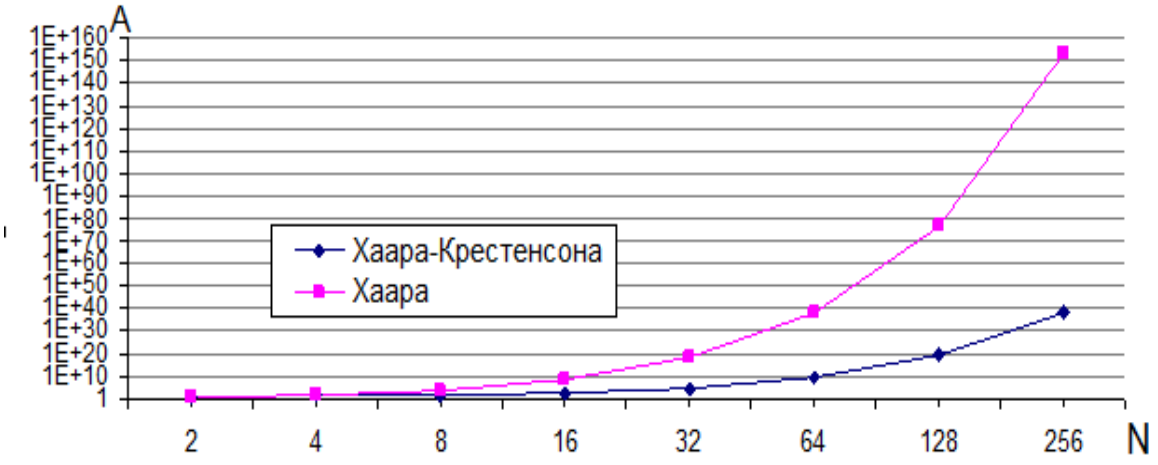


Рисунок 3.7 - Кількість вентильних елементів для базису Хаара та Хаара-Крестенсона

Модульна матриця множення у базисі Хаара-Крестенсона реалізується згідно аналогічної структури з часовою складністю $\tau = \nu$ та меншою апаратною складністю без елементів обчислення рангів $A = p^2$.

3.3.2. Пірамідальні та конвеєрні компоненти спецпроцесорів міжбазисного перетворення Радемахера-Крестенсона

Алгоритми міжбазисного перетворення Радемахера-Крестенсона можуть бути побудовані на основі пірамідальних та конвеєрних структур. У першому випадку виконується паралельне багатокаскадне обчислення залишків, число тактів якого залежить від розрядності вхідних двійкових кодів. У другому випадку відбувається послідовне обчислення залишків, з числом тактів перетворень, пропорційним розрядності перетворюваних двійкових чисел. При цьому виникають компромісні завдання створення міжбазисних перетворювачів з максимальною швидкодією або мінімальною апаратною складністю.

Таким чином, для перетворення двійкових чисел з базису Радимахера в базис Крестенсона над елементами стрічок матриці, поданої в табл.3.2, виконується наступна операція над залишками [37]:

$$res(b_{n-1,j} + b_{n-2,j} + ... + b_{i,j} + ... + b_{1,j} + b_{0,j}) \bmod P_i \ . \qquad (3.3)$$

Для підвищення швидкості модульної операції (3.3) доцільно застосувати структуру пірамідального алгоритму сумування, згідно рис. 3.8 [37, 116, 121].

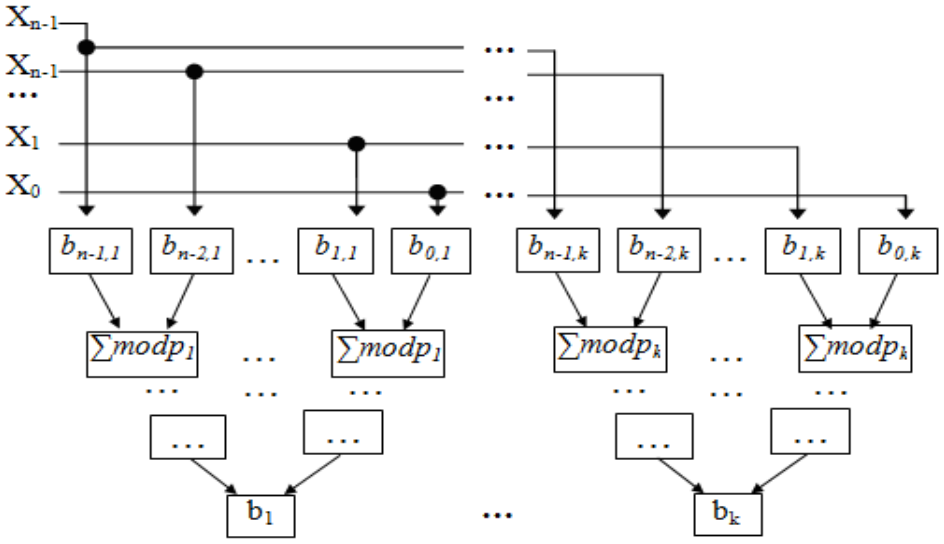


Рисунок 3.8 - Пірамідальний алгоритм сумування залишків в РСЗК

Швидкодія такого пірамідально-модульного суматора розраховується за формулою:

$$m = \log_2 n^i,$$

де n – розрядність процесора базису Радемахера.

Висока швидкодія такого компонента міжбазисного перетворення Радемахера – Крестенсона потребує великої кількості суматорів в залежності від розрядності процесора, число яких розраховується за формулою:

$$S = n + n/2 + n/4 + ... + n/n.$$

Таким чином, загальний об'єм мікроелектронного обладнання даного міжбазисного перетворення можна оцінити згідно виразу

$$Q = K \cdot S,$$

де K – число взаємопростих модулів базису Крестенсона.

Об'єм мікроелектронного обладнання міжбазисного перетворювача (МБП) можна суттєво зменшити на основі інформаційної технології та структури, показаної на рис.3.9 [37, 116, 120, 121].

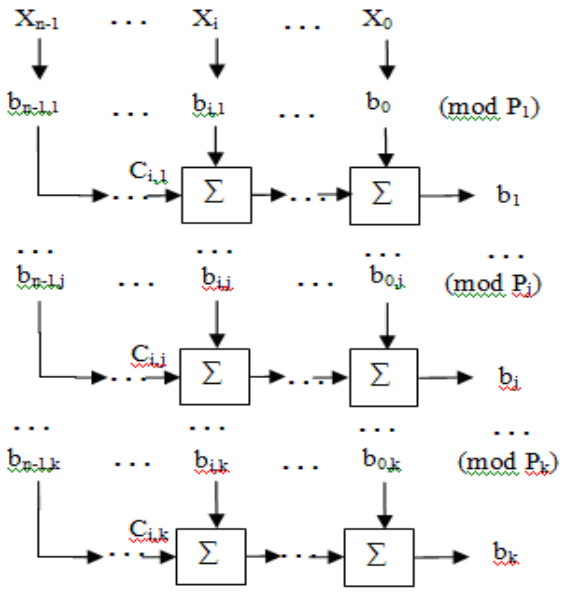


Рисунок 3.9 - Структура міжбазисного перетворювача Радемахера-Крестенсона

Характеристика об'єму та швидкодії мікроелектронного обладнання суматора по модулю Pj. Згідно архітектури пірамідального та лінійного міжбазисних перетворень по модулю P_j, показаних на рис.3.8 та 3.9, порівняння

їх характеристик приведено на рис.3.10, звідки видно, що лінійна архітектура потребує в два рази менше обладнання по відношенню до пірамідальної.

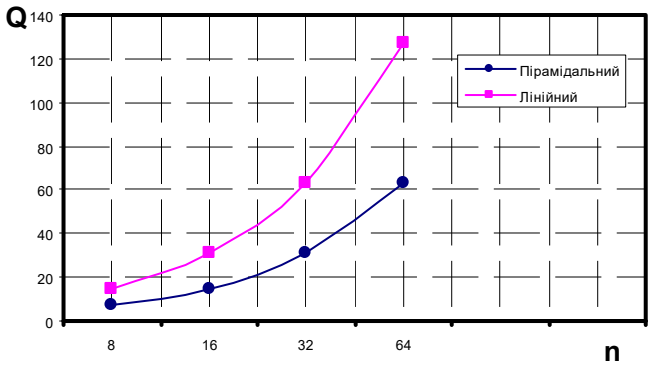


Рисунок 3.10 - Об’єм мікроелектронного обладнання міжбазисного перетворення Радемахера-Крестенсона

Результати аналізу швидкодії двох досліджуваних архітектур міжбазисного перетворення при унітарному кодуванні залишків (рис.3.11) розраховується згідно виразів [37, 121]:

$$S_p = \frac{1}{2+\log_2 n};\ S_l = \frac{1}{n},$$

де n – число розрядів процесора;

S_p – швидкодія пірамідального МБП;

S_l – швидкодія лінійного МБП.

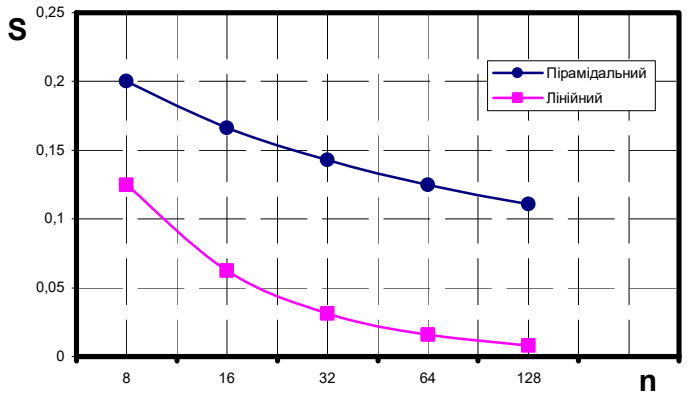


Рисунок 3.11 - Швидкодія міжбазисного перетворення Радемахера-Крестенсона

Проведені дослідження доводять, що в бінарно-розмежованій СЗК підвищується ефективність і реалізація міжбазисного перетворення Радемахера – Крестенсона за схемотехнічними варіантами на основі пірамідально та лінійно з'єднаних суматорів по модулю Р, при чому, об'єм обладнання пірамідальної структури в два рази перевищує об'єм лінійної структури при заданій розрядності процесора. Швидкодія пірамідального МБП зростає із збільшенням розрядності процесора (8 – 128) відповідно у діапазоні (1,6 – 13,8) разів.

При розрядності процесорів k≥1024 застосування досліджуваних міжбазисних перетворень не забезпечує прийнятної швидкодії та апаратної складності, що обмежує застосування цих структур при створенні процесорів шифрування. Крім того, наявність операції порівняння чисел у алгоритмах, які реалізують досліджувані структури, також значно знижує швидкодію міжбазисних перетворювачів.

3.4 Розробка методу та способу визначення залишків на основі модуля пам'яті та мультиплексованих рандомізаторів

Вперше розроблений метод знаходження залишку великорозрядного двійкового числа Y по великорозрядному цілочисельному модулю P, який представляється у доповнюючому коді для виконання операції віднімання, ґрунтується на основі розробленого рекурсивного співвідношення [96, 111]:

$$b_i = [p]_{мд} + 2b_{i-1} + a_i, \ \ i = n, n-1, ..., 1,$$

де n – розрядність числа Y, з якого визначається залишок b_i;

a_i – біти двійкового числа Y, починаючи зі старшого розряду a_n;

$[P]_{мд}$ – $k+1$ розрядна мантиса доповнюючого коду модуля P;

b_i – текуче кодове значення залишку ($b_{i-1} = 0$).

Розглянемо приклад знаходження результату операції за модулем:

Нехай значення Y=$100_{(10)}$=$1100100_{(2)}$, P=$11_{(10)}$=1011.

Отже, залишок в десятковій системі $b_1 = res100(\text{mod}11) = 1$.

Для знаходження значення b_1, знаючи двійкові представлення Y та P, потрібно виконати наступну послідовність кроків:

1. Записуємо число $Y=1100100_{(2)}$ у перший регістр 1;

2. Записуємо $k+1$ розрядну мантису доповнюючого коду $[P]_{мд}=10101$ у другий регістр 5, яку отримуємо наступним чином:

- Двійкове представлення числа P записуємо з нульовим бітом у старшому розряді і отримуємо: $P=01011_{(2)}$
- Записуємо інверсний код для числа P: $\overline{P}=10100$
- Додаємо до цього коду «1»:

$$[P]_{мд}=\begin{array}{r}10100\\ +1\\ \hline 10101\end{array}.$$

3. У регістри 6 і 7 записуємо нулі, тобто $b_i=00000$ $b_{i-1}=00000$; $i=n,n-1,...,1$.

4. У блоці управління формується «0» біт на керуючий вхід мультиплексора, за допомогою якого записується, зчитується інформація b_i, b_{i-1} з регістрів та записуються порозрядно вихідні коди суматора у четвертий регістр.

5. В кожному циклі роботи пристрою виконується сумування мантиси доповнюючого коду модуля P із значенням текучого залишку b_{i-m}. У нашому випадку має місце така операція

$$\begin{array}{ll}[p]_{мд} & 10101\\ 2b_i+a_i & {}^{+}00001\\ \hline [b_{i-m+1}]_{мд} & 11110\end{array}\qquad i=n-1$$

6. В результаті чого отримується мантиса додавання в доповнюючому коді. При чому, записується у регістр третій (b_i) або четвертий (b_{i-1}) двійкове представлення цієї мантиси. Отримане значення «1» в $k+1$ такті сумування показує, що $b_{i-1}<P$. Тоді відбувається зсув інформації у відповідному третьому або четвертому регістрі відбувається запис у молодший розряд a_{i-1} біта числа Y, тобто

$$P > b_i \qquad \begin{array}{r} 10101 \\ {}^{+}00011 \\ \hline 11100 \end{array} \qquad i = n-2;\ P > b_i \qquad \begin{array}{r} 10101 \\ {}^{+}00110 \\ \hline 11011 \end{array} \qquad i = n-3;$$

$$P \le b_i \qquad \begin{array}{r} 10101 \\ {}^{+}01100 \\ \hline 00011 \end{array} \qquad i = n-4.$$

7. Згенерований біт «0» подається на блок управління, в якому він записується і переключається мультиплексор, в результаті чого отримуємо $2b_{i-1}$ в четвертому регістрі та відбувається запис в молодший розряд текучого біта числа *Y*.

8. На основі рекурентного співвідношення здійснюється сумування:

$$\begin{array}{r} 10101 \\ {}^{+}00011 \\ \hline 10100 \end{array} \rightarrow \begin{array}{r} 10101 \\ {}^{+}00110 \\ \hline 11011 \end{array} \rightarrow \begin{array}{r} 10101 \\ {}^{+}01100 \\ \hline 00001 = b_1 \end{array}$$

Отже, на основі використання розробленого методу знаходження залишку отримуємо значення $b_1 = 00001_{(2)} = 1_{(10)}$, що відповідає $b_1 = res100(\mathrm{mod}11) = 1$.

Аналогічним чином виконується розрахунок знаходження залишку по парному модулю:

Нехай $Y=25_{(10)}=11001_{(2)}$, $p=6_{(10)}=110$.

Записуємо $[P]_{мд} = 0110 \xrightarrow{\bar{p}} 1001 \xrightarrow{+1} 1010$.

Згідно запропонованого методу виконуємо наступну послідовність дій:

$$\begin{array}{r} 1010 \\ {}^{+}0001 \\ \hline 1011 \end{array} \rightarrow \begin{array}{r} 1010 \\ {}^{+}0011 \\ \hline 1101 \end{array} \rightarrow \begin{array}{r} 1010 \\ {}^{+}0110 \\ \hline 0000 \end{array}$$

Слід зазначити, якщо старший розряд «0», то здійснюється зсув на біт отриманого залишку і додається новий біт числа:

$$\begin{array}{r} 1010 \\ {}^{+}0000 \\ \hline 1010 \end{array} \rightarrow \begin{array}{r} 1010 \\ {}^{+}0001 = b_1 \\ \hline 1011 \end{array}$$

Отже, коли використані всі біти числа Y і $b_i < P$, то $b_1 = 0001_{(2)}$, то $b_1 = res25(\mathrm{mod}6) = 1$.

При реалізації пристрою доцільно в якості регістрів використати багаторозрядну флеш-пам'ять.

Функціональним обмеженням розробленого спецпроцесора є наявність операцій додавання доповнюючих кодів фрагментів двійкового числа базису Радемахера для визначення залишку по модулю p_i, що знижує швидкодію роботи процесора. Тому, з метою підвищення швидкодії спецпроцесорів даного класу, запропонована структурна схема на основі використання ПЗП, в якому зберігаються обчислювані залишки по модулю.

В основу способу отримання залишку b_i двійкового числа $X_{(2)} = (a_0, a_1, ...a_i ...a_{n-1})$, який починаючи з його старшого розряду a_{n-1}, по заданому модулю P_j, покладено рекурентний алгоритм [97, 98, 111].

$$b_i = (a_i + 2 \cdot b_{i-1}) \bmod P_j, \tag{3.4}$$

де a_i – значення i-того біта двійкового числа;

b_{i-1}– значення залишку $i-1$ біта двійкового числа. Початкова умова реалізації способу визначення залишку задається наступними даними: $i = n-1$, $b_{i-1} = 0$. Шуканий кінцевий залишок b_0 отримується згідно виразу:

$$b_0 = resX(\mathrm{mod}\, P_j),$$

де *res*- символ операції визначення найменшого невід'ємного залишку.

Наприклад, потрібно обчислити залишок числа $X=100_{(10)}=1100100_{(2)}$ по модулю $P=11_{(10)}=1011_{(2)}$:

У регістр пам'яті та зсуву в *n*-розрядів записуємо X=1100100, а в інші *m*-розряди b_{i-1}=00000.

(Зсув 1) Зсуваємо код числа X, починаючи з старшого розряду вправо, при цьому, старший розряд числа X потрапляє в перший розряд *m*-розрядного регістра пам'яті та зсуву, тобто при вхідному коді b_{i-1}=10000 на виході відповідно отримається b_i=10000 (рис.3.12), який порозрядно записується в *m*-розряди регістра пам'яті та зсуву.

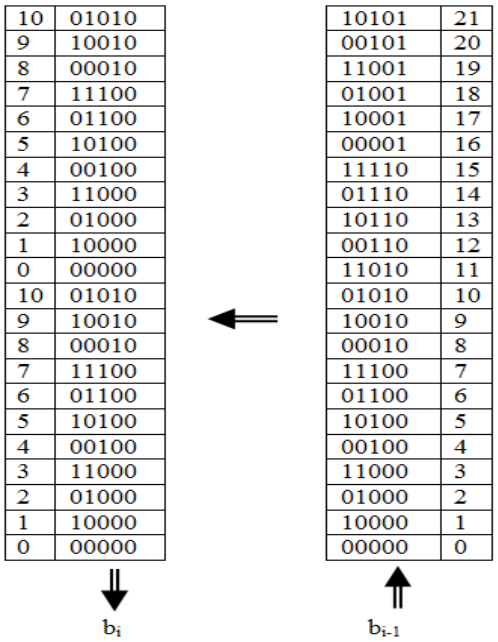


Рисунок 3.12 - Приклад вибірки кодів залишків b_i згідно адресних входів b_{i-1} по модулю $P_{(10)}=11$ з постійним запам'ятовуючим пристроєм

Аналогічно здійснюються наступні зсуви, при яких b_{i-1} та b_i будуть набувати значень (див. рис.3.12):

(Зсув 2) b_{i-1}=11000, b_i=11000;

(Зсув 3) b_{i-1}=01100, b_i=01100;

(Зсув 4) b_{i-1}=00110, b_i=10000;

(Зсув 5) b_{i-1}=11000, b_i=11000;

(Зсув 6) b_{i-1}=01100, b_i=01100;

(Зсув 7) b_{i-1}=00110, b_i=10000.

Після n=7 зсувів отримаємо кінцеве значення залишку b_0=10000, що зчитується вихідною шиною, в якому старший розряд справа.

Операція обчислення залишку даним способом потребує *n* тактів в регістрі пам'яті та зсуву та *n* тактів вибірки коду залишку з постійного запам'ятовуючого пристрою, тобто рівне *2n*, у випадку, якщо двійкове число буде займати 512 біт, а модуль, за яким обчислюють залишок, буде від 2 біт до 128, то швидкодія, в порівнянні з відомим способом, зросте від 2-х до 48-и разів.

Функціональним обмеженням дослідженого алгоритму та розробленої структурної схеми спецпроцесора є лінійне зростання часової складності виконання операції міжбазисного перетворення Радемахера-Крестенсона пропорційна розрядності двійкових чисел базису Радемахера, що обмежує можливості його використання при опрацюванні великорозрядних чисел.

З метою створення спецпроцесора міжбазисного перетворення Радемахера-Крестенсона з максимальною швидкодією, незалежно від розрядності вхідних двійкових чисел, розроблений рекурентний алгоритм, який описується виразом [99, 111, 112]

$$b_i = (((((((1 \cdot 2 + a_{n-1}) \bmod p_i) \cdot 2) \bmod p_i + a_{n-2}) \bmod p_i \cdot 2) + ...$$
$$... + a_1) \cdot 2) \bmod p_i + a_0) \bmod p_i .$$

Поставлена задача підвищення швидкодії та зменшення апаратної складності вирішується завдяки тому, що пристрій для перетворення чисел з позиційної системи числення в систему залишкових класів містить шини для подачі K –розрядного позиційного числа, перетворювачі степенів розрядів числа в позиційній системі числення по модулю P_j, виходи кодів залишків системи залишкових класів і K комутаційних мультиплексорів по кожному модулю системи залишкових класів P_j, перші входи яких підключені до відповідних шин вхідного K –розрядного двійкового числа, а другі входи i-тих комутаційних мультиплексорів підключені до виходів i-1-ших комутаційних мультиплексорів, починаючи з K-2 до нульового розряду, при чому, на одиничний вхід K-1-го комутаційного мультиплексора подається старший біт двійкового числа базису Радемахера, а виходи нульового комутаційного мультиплексора є виходами чисел залишків b_i по модулю P_j системи залишкових класів.

Функціональна структура одного розряду міжбазисного перетворювача показана на рис. 3.13(а), що складається з *Rand* – рандомізатор по модулю, *Incr* – інкрементний пристрій, *MX* - P-канальний двохвходовий мультиплексор.

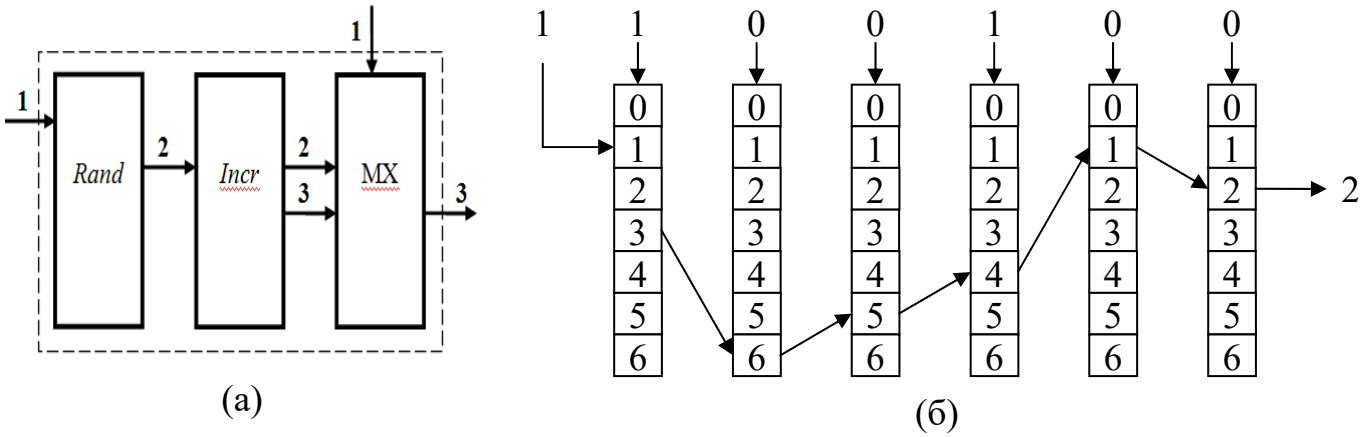
Наприклад, потрібно обчислити залишок числа $X=100_{(10)}=1100100_{(2)}$ по модулю $P=7_{(10)}=111_{(2)}$, який показаний у вигляді графа на рис.3.13(б):

– на вхідну шину подається двійкове значення числа $X=1100100$, старший біт, якого приймаємо за перший залишок $X_6=b_1=1$, що поступає на вхід рандомізатора по модулю $P=7$;

– в результаті, на виході з врахуванням старшого біта «1», згідно виразу (3.4), отримується залишок рандомізатора $b_{1,Rand}=2$;

– отримане $b_{1,Rand}$ поступає на інкрементний пристрій, на виході якого отримується два значення $b_{1,Incr}=2;3$, що поступають на відповідні входи P-канального двохвходового мультиплексора. На другий вхід P-канального двохвходового мультиплексора поступає з вхідної шини наступний біт $X_5=1$, який встановлює його у відповідний режим роботи.

В результаті, з мультиплексора кожного розряду отримується наступний залишок $b_{1,MX}= b_2=3$.



(а) (б)

Рисунок 3.13 – Структура одного розряду міжбазисного перетворювача (а) та граф обчислення залишку (б) числа $X=100_{(10)}=1100100_{(2)}$

Аналогічно, паралельно виконуються операції для інших розрядів двійкового числа, де вхідні дані b_i та X_i, вихідні дані $b_{i,Rand}$, $b_{i,Incr}$ та $b_{i,MX}= b_{i+1}$:

– вхідні дані: $b_2=3$, $X_4=0$; вихідні дані: $b_{2,Rand}=6$, $b_{2,Incr}=6;0$ та $b_{2,MX}=b_3=6$;

– вхідні дані: $b_3=6$, $X_3=0$; вихідні дані: $b_{3,Rand}=5$, $b_{3,Incr}=5;6$ та $b_{3,MX}=b_4=5$;

– вхідні дані: $b_4=5$, $X_2=1$; вихідні дані: $b_{4,Rand}=3$, $b_{4,Incr}=3;4$ та $b_{4,MX}=b_5=4$;

– вхідні дані: $b_5=4$, $X_1=0$; вихідні дані: $b_{5,Rand}=1$, $b_{5,Incr}=1;2$ та $b_{5,MX}=b_6=1$;

– вхідні дані: $b_6=1$, $X_0=0$; вихідні дані: $b_{6,Rand}=2$, $b_{6,Incr}=2;1$ та $b_{6,MX}=b_7=2$.

Кінцевим результатом обчислення залишку числа $X=100$ по модулю $P_j=7$ буде $b_j=b_7=2$.

3.5 Розробка функціональної структури визначення кодів взаємопростих залишків СЗК

Для отримання простих чисел, які формують приведені таблиці ділимості чисел (див. табл.2.9-2.13), розроблена функціональна структура компонента процесора в РСЗК, яка зображена на рис.3.14 [37].

Пристрій містить наступні компоненти функціональні компоненти:

RP – регістр доповнюючого коду модуля p, який представляється в нормалізованому вигляді у доповнюючому коді, згідно виразу: $[p]_{\partial} = 1,(\overline{p} + 1 \cdot 2^{0})$;

RB – регістр поточного залишку b_i, реалізований на D-тригерах на виходах якого формуються біт орієнтовані двійкові коди залишку починаючи з молодших розрядів, які поступають на суматор ∑ та модуль порівняння МР;

RZ – регістр подвоєння коду залишку, який формується у випадку, коли $b_i<p$;

∑ – двійковий багаторзрядний суматор на базі однорозрядних повних суматорів, який виконує обчислення залишку суми $b_{i+1} = [p]_{\partial} + b_i$, код якого формується у нормалізованому вигляді прямого нормалізованого коду отриманого залишку.

БУ – мікропрограмний блок управління.

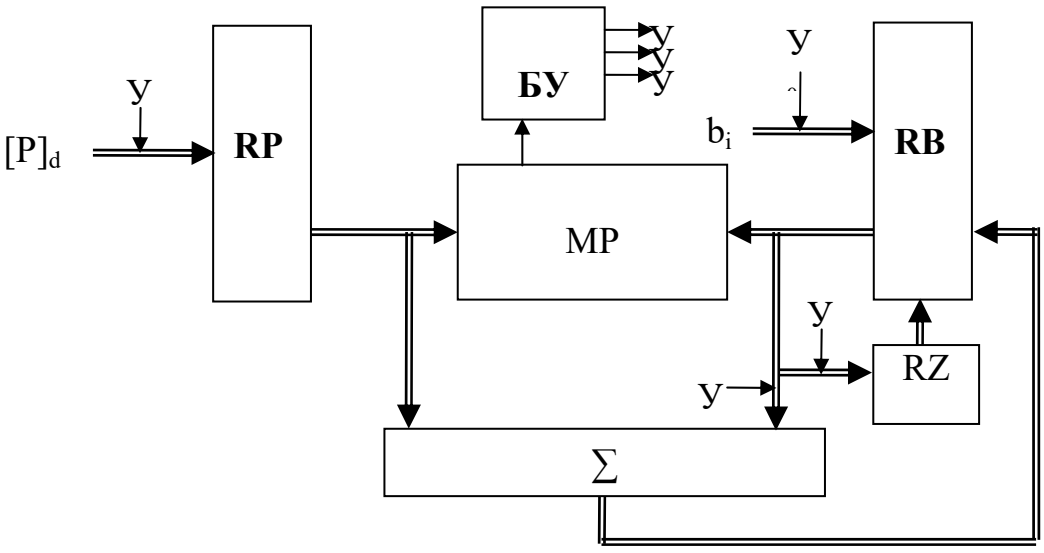


Рисунок 3.14 - Функціональна структура компонента процесора РСЗК

Базовим компонентом даного процесора є виконання операції порівняння двох чисел $[P]_d<b_i$, з метою спрощення даного модуля операція порівняння виконується шляхом додавання доповнюючого коду модуля до $[P]_d$ коду поточного залишку. При цьому, якщо $b_i < [P]_d$ то на виході сумматора $\sum$ формується біт «1», в протилежному випадку – біт «0».

Пристрій працює наступним чином.

В регістри RP та RB записуються, відповідно, доповнюючий код числа $[P]_d$, де P – певне число бітів, та b_i – залишок числа P. Записані коди порівнюються між собою, в разі виконання умови $[P]_d<b_i$ дані регістрів подаються на суматор, інакше відбувається зсув вправо коду числа b_i, який записується в регістр RB, поки умова не буде виконуватись. Робота такого процесора реалізується згідно нищі аналогічної оцінки мікрокоманд блоку управління, який керує процесом руху даних. Такий БУ контролює три стани:

$У_0$ – запис даних в регістри RP:= $[P]_d$, RB:= b_i;

$У_1$ – зсув коду числа b_i , RZ:=2 RZ;

$У_2$ – сумування даних регістрів RP та RB , $\sum$:=$[P]_d$+ b_i.

Розроблена функціональна структура пристрою визначення кодів взаємопростих залишків СЗК застосований при розробці спецпроцесора визначення залишків багаторозрядних двійкових чисел на основі одно-розрядного повного суматора (патент України № 68872 [91]).

ВИСНОВКИ ДО РОЗДІЛУ 3

1. Розроблені принципи та теоретичні основи розмежування розрядної сітки у базисі Радемахера-Крестенсона у результаті чого отримані аналітичні вирази прямого та зворотнього перетворень розмежованої системи. Показано, що при бінарному розмежуванні базису Радемахера формуються матриці степеневих залишків двійкових чисел, які дозволяють зменшити на 2-3 порядки обчислювальну та часову складність міжбазисних перетворень, а також операцій додавання, множення та піднесення до степеня великорозрядних двійкових чисел.

2. Отримані аналітичні вирази та досліджена часова складність алгоритмів міжбазисних перетворень Радемахера-Крестенсона та Крестенсона-Радемахера на основі цілочисельної, нормалізованої, досконалої та розмежованої форм СЗК, а також, міжбазисних перетворень Радемахера-Галуа, Крестенсона-Галуа та Галуа-Радемахера і Крестенсона-Галуа. Розрахована та побудована діаграма обчислювальної складності досліджених міжбазисних преретворень. В результаті виконаних досліджень встановлено найменшу обчислювальну складність досконалих форм СЗК, які на порядок нищі аналогічної оцінки цілочисельної СЗК та на 1-2 порядки менші обчислювальної складності перетворень Радемахера, Галуа, Крестенсона.

3. Розроблені теоретичні засади та алгоритми виконання міжбазисних перетворень на основі матричних суматорів у розмежованій СЗК пірамідального та лінійного типу. Розроблені структури такого класу суматорів та оцінені характеристики часової та апаратної складності в залежності від розрядності чисел базису Радемахера, в результаті встановлено, що в бінарно-розмежованій СЗК підвищується ефективність реалізації міжбазисного перетворення Радемахера – Крестенсона за схемотехнічними варіантами на основі пірамідально та лінійно з’єднаних суматорів по модулю Р, при чому, об’єм обладнання пірамідальної структури в два рази перевищує об’єм лінійної структури при заданій розрядності процесора. Швидкодія пірамідального МБП

зростає із збільшенням розрядності процесора від 8 – 128 відповідно у діапазоні 1,6 – 13,8 разів.

4. Запропоновані рекурентні алгоритми та високопродуктивні спецпроцесори опрацювання великорозрядних чисел у базисі Радемахера-Крестенсона шляхом використання однорозрядного двійкового суматора зі схемою порівняння на основі операції додавання доповнюючих кодів мантиси моулів та фрагментів двійкового числа базису Крестенсона та застосування ПЗП в якості обчислювача текучих сум залишків по модулю, що дозволило зменшити на порядок апаратну складність та підвищити на 2 порядки швидкодію спецпроцесорів даного класу.

РОЗДІЛ 4
РОЗРОБКА ТА РЕАЛІЗАЦІЯ СПЕЦПРОЦЕСОРІВ ОПРАЦЮВАННЯ ВЕЛИКОРОЗРЯДНИХ ЧИСЕЛ У БАЗИСІ КРЕСТЕНСОНА

4.1 Розробка та дослідження алгоритмів піднесення до високих показників степенів у РСЗК базису Крестенсона

Операція модулярного експоненіювання $a^x mod\ p$ (крім того, $x \le \varphi(p)$, $\varphi(p)$ – значення функції Ейлера від модуля p) є однією з базових в асиметричних алгоритмах шифрування RSA, Ель–Гамаля [94]. Для виконання даної операції потрібно використати матрицю, представлену у вигляді таблиці 4.1, розмірність якої співмірна з розрядністю p, тобто рівна n. При чому, в стовпцях таблиці подані заначення $a^2 \bmod p$ в базисі Радемахера, a_{ij}=0,1. Для зменшення часової складності, степінь x записуємо степенями 2, і результат операції модулярного експоненціювання отримується шляхом перемноження відповідної кількості стовпців даної таблиці 4.1 з використанням методу модулярного множення в ТЧБ Радемахера-Крестенсона.

Таблиця 4.1 - Матриця піднесення до степеня в базисі Радемахера–Крестенсона

$a_{n-1\ n-1}$	…	$a_{i\ n-1}$		$a_{1\ n-1}$	$a_{0\ n-1}$
…	…	…	…	…	…
$a_{n-1\ j}$	…	$a_{i\ j}$	…	$a_{1\ j}$	$a_{0\ j}$
…	…	…	…	…	…
$a_{n-1\ 1}$	…	$a_{i\ 1}$	…	$a_{1\ 1}$	$a_{0\ 1}$
$a_{n-1\ 0}$	…	$a_{i\ 0}$	…	$a_{1\ 0}$	$a_{0\ 0}$
$a^{2^{n-1}}$	…	a^{2^i}	…	a^{2^1}	a^{2^0}

В запропонованому алгоритмі модулярного експоненціювання, в якому на відміну від відомих, здійснюється заміна операції множення великорозрядних чисел операцією сумування, що дозволяє зменшити обчислювальну складність та збільшити швидкодію на 30-40% для параметрів, менших 256 біт (рис.4.1), на 10% - більших від 256 біт. Результати досліджень показали, що розроблений метод дозволяє зменшити складність з $O3 = \frac{n^3}{2}$ до

$$O4(n)=\begin{cases}\log_2 n\cdot\left(\frac{n}{2}\cdot\log_2 n+1\right), якщо\ n<256\\ n\cdot\left(\frac{n}{2}\cdot\log_2 n+1\right), в\ інших\ випадках\end{cases}$$ в $$E2(n)=\begin{cases}\frac{n}{(\log_2 n)^2+2\cdot\log_2 n}, для\ n<256\\ \frac{n}{(\log_2 n)+2\cdot\log_2 n}, 256\le n\le 1024\end{cases}$$

разів, що показано на рис. 4.2 [87].

В результаті проведеного чисельного експерименту можна зробити висновок, що розроблений метод є ефективним та високопродуктивним у задачах обчислення значення модулярного експоненціювання будь-якої розрядності. Крім того, при зростанні розрядності вхідних параметрів операції модулярного піднесення до степеня спадає ефективність, оскільки частина ресурсу комп'ютера буде задіяна на розв'язок службової інформації, тому значно зростає складність. Отже, даний метод доцільно застосовувати в асиметричних системах захисту інформаційних потоків для зменшення обчислювальної складності, і відповідно, нарощування рівня захисту за рахунок збільшення розрядності вхідних параметрів.

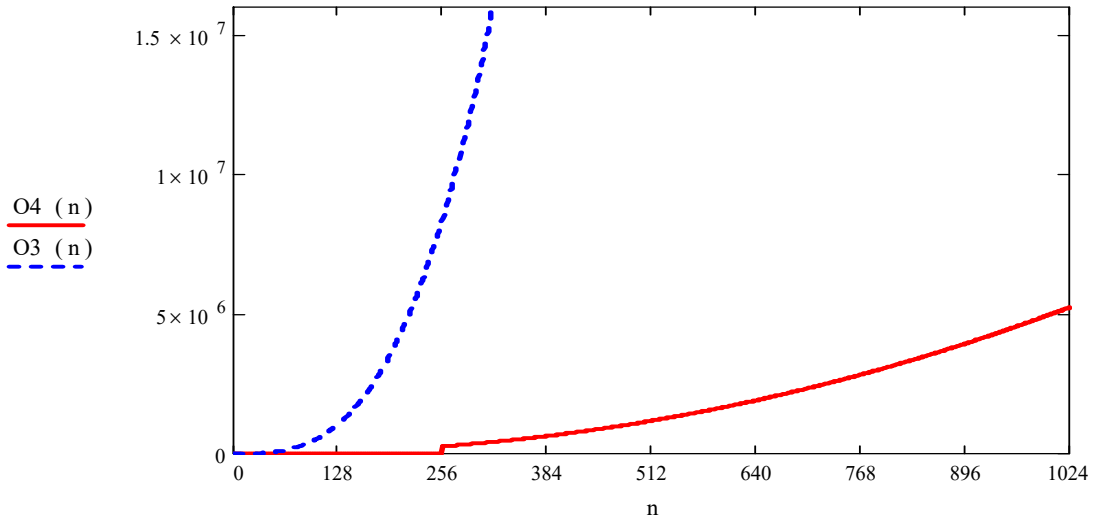


Рисунок 4.1 - Обчислювальна складність операції модулярного піднесення до степеня

В умовах стрімкого розвитку комп'ютерної техніки зростають вимоги до забезпечення необхідного рівня захищеності даних та додаткового опрацювання інформаційних потоків в комп'ютерних мережах. Тому результати проведеного аналізу ефективності досліджуваних методів показали, що застосуванням теоретичних засад пропонованих підходів дає змогу

вдосконалити захист даних на основі використання асиметричних криптоалгоритмів.

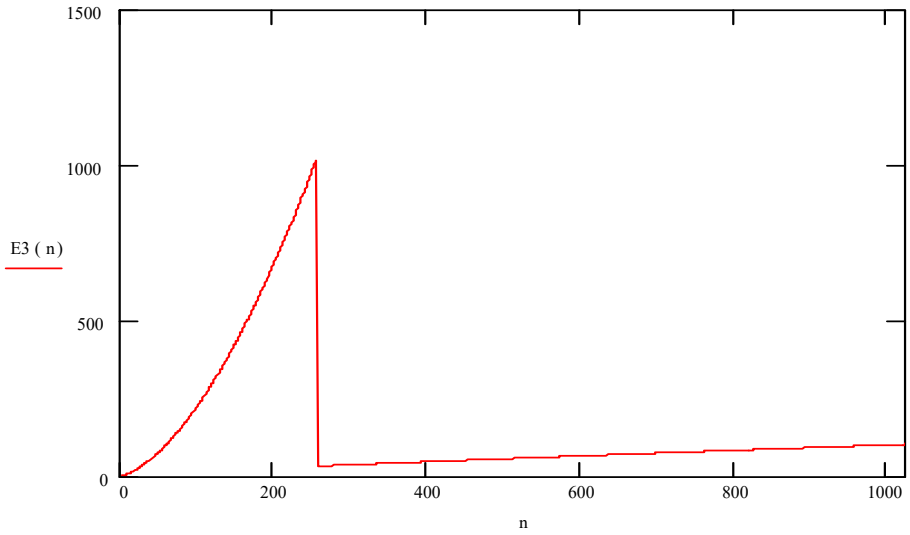


Рисунок 4.2 - Ефективність запропонованого алгоритму

Криптографічна система з відкритим ключем RSA ґрунтується на однонаправлених задачах факторизації чисел великої розрядності, які є обчислювально складними. Алгоритм RSA складається з трьох основних етапів, а саме: генерування ключів, шифрування та дешифрування. Оскільки RSA належить до двоключової криптографії, то адресат і отримувач в процесі шифрування мають в своєму розпорядженні відповідно відкритий і таємний ключі, для генерування яких потрібно здійснити наступні кроки [87]:

1. Вибираються випадковим чином два прості числа p і q будь-якої розрядності, при чому розмірність впливає на стійкість системи захисту від несанкціонованого доступу, тобто зі збільшенням розрядності параметрів зростає стійкість, яка ґрунтується на задачі факторизації.

2. Знаходимо значення $n = p * q$ – модуль криптоперетворень на основі використання матричних перетворень. Для цього представляємо p і q у вигляді: $p = p_{r-1}2^{r-1} + p_{r-2}2^{r-2} + \ldots + p_1 2^1 + p_0 2^0$, $q = q_{r-1}2^{r-1} + q_{r-2}2^{r-2} + \ldots + q_1 2^1 + q_0 2^0$, де r - розрядність чисел p і q. Знаходимо значення операції множення з використанням таблиці 4.2, де $m_{ij} = 2^{i+j}$.

Множення чисел p і q здійснюється згідно співвідношення:

$$n = p \cdot q = \sum_{s,k=1}^{r-1} m_{sk}\ , \qquad (4.1)$$

де $p_s, q_k = 1$, тобто m_{sk} знаходиться на перетині стовпця та рядка, для яких відповідні p_i і q_j дорівнюють 1. Використання даного підходу дозволяє зменшити обчислювальну складність модуля криптоперетворень $O(3{,}5n^2 + n - 1/2)$.

Таблиця 4.2 - Матриця знаходження модуля перетворення в базисі Радемахера–Крестенсона

	q_{r-1}	…	q_j	…	q_1	q_0
p_{r-1}	$m_{r-1\ r-1}$	…	$m_{r-1\ j}$	…	$m_{r-1\ 1}$	$m_{r-1\ 0}$
…	…	…	…	…	…	…
p_i	$m_{i\ r-1}$	…	m_{ij}	…	m_{i1}	m_{i0}
…	…	…	…	…	…	…
p_1	$m_{1\ r-1}$	…	m_{1j}	…	m_{11}	m_{10}
p_0	$m_{0\ r-1}$	…	m_{0j}	…	m_{01}	m_{00}

3. Отримавши значення модуля криптоперетворення, аналогічним чином знаходимо $\varphi(n) = (p-1)(q-1)$.

4. Вибирається ціле число $e, \_1 < e < \varphi(n)$, таким чином, щоб $НСД(e, \varphi(n)) = 1$. Для забезпечення необхідного рівня захисту дану експоненту доцільно вибирати числа Ферма 17, 257, 65537… , тобто з найменшою кількістю одиничних бітів в двійковій формі. Задане число e називається відкритим ключем, при чому із зменшенням розмірності e знижується стійкість криптографічного алгоритму RSA.

5. Для знаходження закритого ключа d потрібно знайти значення оберненого до елемента e за модулем $\varphi(n)$, тобто $d \equiv e^{-1} (\mathrm{mod}\, \varphi(n))$. Відомі методи пошуку оберненого елемента за модулем використовують алгоритм Евкліда та його наслідок. Щоб зменшити обчислювальну складність, можна скористатися матричним методом, який передбачає наступну послідовність дій:

Припустимо, що x пробігає зведену систему залишків за модулем $\varphi(n)$: $r_1 = 1, r_2 = 2, r_3 = 3, ..., r_i = i, ..., r_{\varphi(n)-1} = \varphi(n) - 1$. Відповідно, добуток $e \cdot x$, якщо $e < \varphi(n)$, також пробігатиме зведену систему залишків:

$$\begin{cases} e \cdot r_1 \bmod \varphi(n) = c_1 \\ e \cdot r_2 \bmod \varphi(n) = c_2 \\ e \cdot r_3 \bmod \varphi(n) = c_3 \\ \quad \ldots\ldots\ldots\ldots\ldots\ldots \\ e \cdot r_i \bmod \varphi(n) = c_i \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ e \cdot r_{\varphi(n)-1} \bmod \varphi(n) = c_{\varphi(n)-1}. \end{cases} . \quad (4.2)$$

Аналогічним чином для d :

$$\begin{cases} d \cdot r_1 \bmod \varphi(n) = g_1 \\ d \cdot r_2 \bmod \varphi(n) = g_2 \\ d \cdot r_3 \bmod \varphi(n) = g_3 \\ \quad \ldots\ldots\ldots\ldots\ldots\ldots \\ d \cdot r_i \bmod \varphi(n) = g_i \\ \ldots\ldots\ldots\ldots\ldots\ldots \\ d \cdot r_{\varphi(n)-1} \bmod \varphi(n) = g_{\varphi(n)-1}. \end{cases} . \quad (4.3)$$

З системи (4.2) вибираємо $e \cdot r_1 \bmod \varphi(n) = c_1$, з другого – рівняння, в якому $c_1 = r_i$: $e \cdot r_1 \bmod \varphi(n) = c_1$

$$d \cdot r_i \bmod \varphi(n) = g_i . \quad (4.4)$$

Знайшовши добуток, отримаємо:

$$er_1 \cdot dr_i \bmod \varphi(n) = c_1 \cdot g_i . \quad (4.5)$$

Оскільки $r_1 = 1, c_1 = r_i$, то отримується добуток за заданим модулем:

$$e \cdot d \bmod \varphi(n) = g_i . \quad (4.6)$$

6. Запропонований метод дозволяє знаходити добутки великої кількості даних, обернені елементи за модулем та корені квадратні за модулем. Результат даних операцій зручно знаходити за допомогою таблиці Келі (абл.. 4.3), наприклад, для $\varphi(n)$=20, $p = 3, q = 11$.

Знайшовши обернений елемент за модулем $d \equiv e^{-1} (\bmod \varphi(n))$, отримуємо пари $P = (e, n)$ - відкритий ключ, $S = (d, n)$ - таємний ключ.

Для здійснення шифрування, яке ґрунтується на операції модулярного експоненціювання, інформаційний потік $M \in D \in Z_n, 0 < M < n-1$ подається у вигляді цілого числа(порядковий номер літер в алфавіті) і з використанням

відкритого ключа $P=(e,n)$ на підставі співвідношення $P(M)=M^e \bmod n$ отримуємо шифротекст.

Таблиця 4.3 - Таблиця Келі для множення

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

Шифрування даних в інформаційному потоці доцільно здійснювати з використанням розробленого алгоритму модулярного експоненціювання на основі матрично-модульних перетворень ТЧБ Радемахера-Крестенсона. Шифротекст $P(M)=M^e \bmod n$ передається по каналу зв'язку. Для пошуку значення $P(M)$ потрібно побудувати матрицю розмірності n, що дорівнює розрядності модуля p у вигляді таблиці. В стовпцях таблиці записуємо значення $M^{2^i} (\bmod n)$ в двійковій системі числення, тобто M_{ij}=0, 1 (табл.4.4). Згідно розроблених теоретичних положень для методу модулярного експоненціювання будь-який степінь e записується за степенями 2 і результат отримується шляхом перемноження відповідної кількості стовпців табл. 4.4.

Таблиця 4.4 - Матриця шифротексту алгоритму шифрування RSA в базисі Радемахера–Крестенсона

$M_{n-1\ n-1}$	…	$M_{i\ n-1}$		$M_{1\ n-1}$	$M_{0\ n-1}$
…	…	…	…	…	…
$M_{n-1\ j}$	…	$M_{i\ j}$	…	$M_{1\ j}$	$M_{0\ j}$
…	…	…	…	…	…
$M_{n-1\ 1}$	…	$M_{i\ 1}$	…	$M_{1\ 1}$	$M_{0\ 1}$
$M_{n-1\ 0}$	…	$M_{i\ 0}$	…	$M_{1\ 0}$	$M_{0\ 0}$
$M^{2^{n-1}}$	...	M^{2^i}	...	M^{2^1}	M^{2^0}

В основі процесу дешифрування лежить операція модулярного експоненціювання $S(P(M)) = (P(M))^d \bmod n$, тільки з застосуванням таємного ключа $S = (d, n)$ до зашифрованного інформаційного потоку $P(M)$.

Отже, при реалізації криптографічного алгоритму захисту інформаційних потоків RSA доцільно використовувати теоретичні основи ТЧБ Радемахера та Крестенсона, які дозволяють на 1-2 порядки зменшити обчислювальну складність базової $P(M) = M^e \bmod n$. В праці [100] показано, що для реалізації даної операції на основі використання алгоритму швидкого піднесення до степеня потрібно $O(\ln e)$ операцій множення по модулю. Таким чином, взявши до уваги, що модулярне множення має обчислювальну складність $O(r^2)$, то складність $P(M) = M^e \bmod n$ буде $O(n^2 \cdot \ln e)$, де n - розрядність модуля n. Крім того, як було зазначено, в криптоалгоритмі шифрування/дешифрування інформаційних потоків RSA використовується одна і та ж операція модулярного експоненціювання, то обчислювальна складність його буде $O1(n^2 \cdot \ln e + 2n)$, тоді як запропонованого алгоритму з використанням ТЧБ Радемахера-Крестенсона $O3(n) = \begin{cases} \log_2 n\left(3\log_2 n + \frac{n}{2}\right), \textit{якщо } n < 256 \\ n \cdot \left(3\log_2 n + \frac{n}{2}\right), \textit{в інших випадках} \end{cases}$, що показано на рис. 4.3.

Результати досліджень обчислювальних складностей криптоалгоритму шифрування/дешифрування інформаційних потоків RSA в двовимірному ТБЧ

Рademахера та на основі використання запропонованих методів в ТЧБ Рademахера-Крестенсона показує, що застосування розроблених алгоритмів дозволить зменшити складність на 20-40% для параметрів в діапазоні до 256 біт, і на 10 % при параметрах від 256 до 1024 біт.

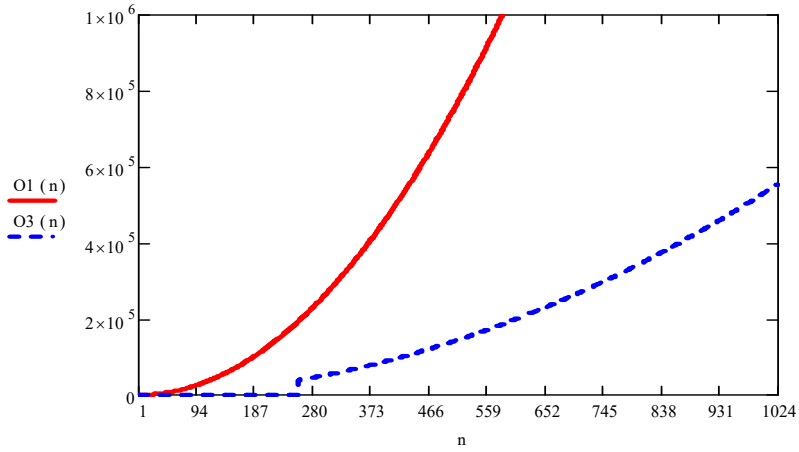


Рисунок 4.3 - Обчислювальні складності: $O3(n)$ - алгоритму шифруваня RSA з використанням розмежованої системи числення Рademахера-Крестенсона, $O1(n)$ - класичного алгоритму RSA

В алгоритмі шифрування Ель-Гамаля для здійснення процесу шифрування/дешифрування необхідно згенерувати ключі, виконуючи наступну послідовність кроків:

1. Задається випадкове просте число p розмірності n.
2. Вибираємо будь-яке $g \in Z$, яке є первісним коренем по модулю p, і довільне число $x \in (1, p)$, таке, що $НСД(x, p-1) = 1$.

Знаходимо відкритий ключ згідно співвідношення $y = g^x mod\ p$ на основі використання розробленого матрично-модульного методу модулярного експоненціювання в ТЧБ Рademахера–Крестенсона. В стовпцях таблиці записується величини $g^{2^i} (\mathrm{mod}\, n)$ в двійковій системі числення, тобто g_{ij}=0, 1(табл. 3.14) та предсталямо степінь x з степенями 2, тобто $x = x_{n-1}2^{n-1} + \ldots + x_i 2^i + \ldots + x_1 2 + x_0$. Значення операції $y = g^x mod\ p$ знаходимо в результаті перемноження відповідної кількості стовбців табл. 3.14. Таким чином, отримуємо відкритий ключ алгоритму шифрування Ель-Гамаля, із значно меншою обчислювальною складністю.

4.2 Розробка та дослідження системних характеристик модульних компонентів спецпроцесорів у базисі Крестенсона

Створення спецпроцесорів опрацювання великорозрядних чисел у базисі Крестенсона потребує схемотехнічної реалізації спеціалізованих компонентів:

– лічильників по модулю P;
– аналого цифрових перетворень базису Крестенсона;
– шифраторів Радемахера-Крестенсона;
– рандомізаторів по модулю p_i.

Вказані модульні компоненти спецпроцесорів досліджуваного класу можуть бути з різною ефективністю реалізовані на основі алгоритмічного та схемо-технічного виконання у різних ТЧБ згідно заданих критеріїв мінімальної апаратної та часової складності в залежності від типу обчислювальних задач.

Тому створення цих компонентів спецпроцесорів у базисі Крестенсона потребує теоретико-математичної формалізації алгоритмів виконання операцій такими цифровими пристроями та оптимізації їх програмно-апаратних, функціонально-структурних та схемо технічних рішень.

4.2.1 Пристрої обчислення залишків на основі лічильників по модулю в різних ТЧБ

Систематизація модульних лічильників у різних ТЧБ, які виконують перетворення чисел з унітарного базису в інші ТЧБ, представлена в табл.4.5, а їх структури – у табл.4.6.

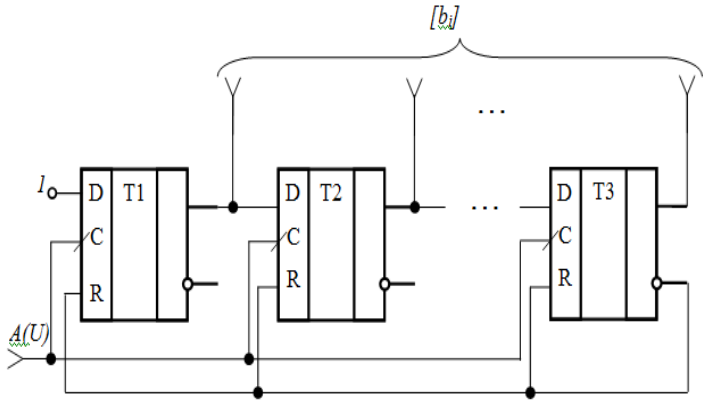
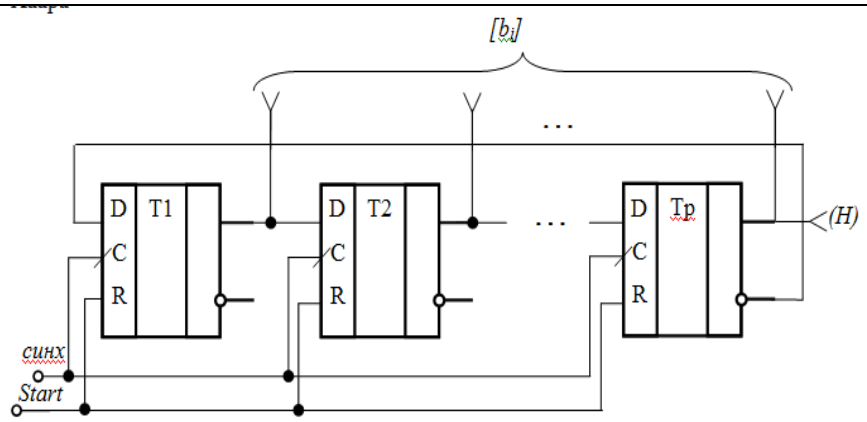
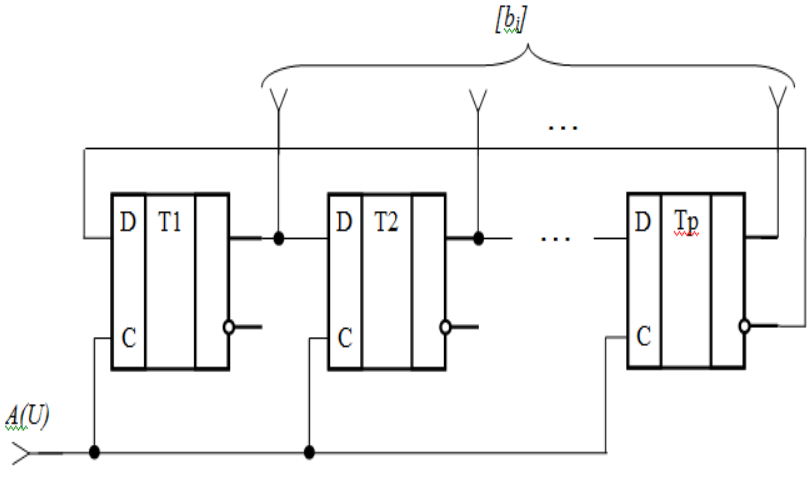
Таблиця 4.5 - Модульні лічильники у різних ТЧБ

ТЧБ	Асинхронні	Синхронні
1	2	3
Унітарний	-	+1
Хаара	-	+1
Крейга	-	+1

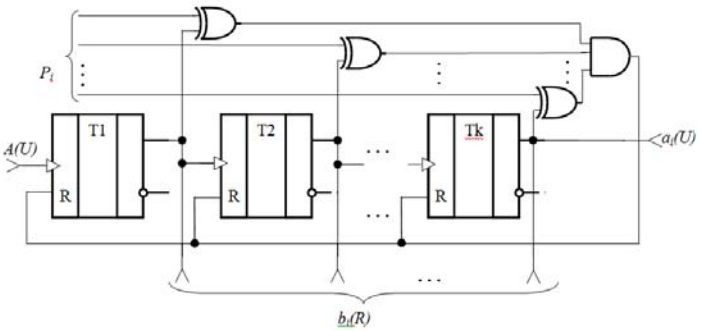
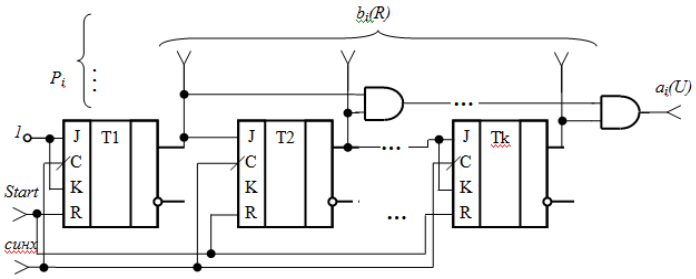
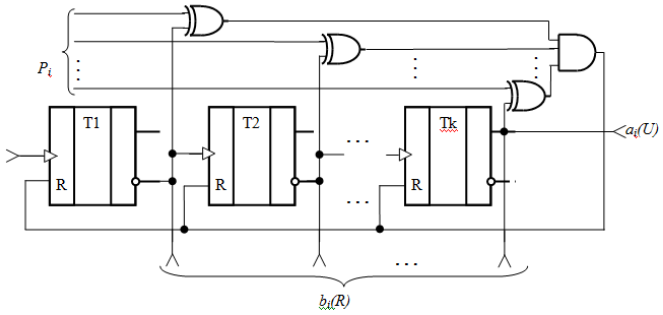
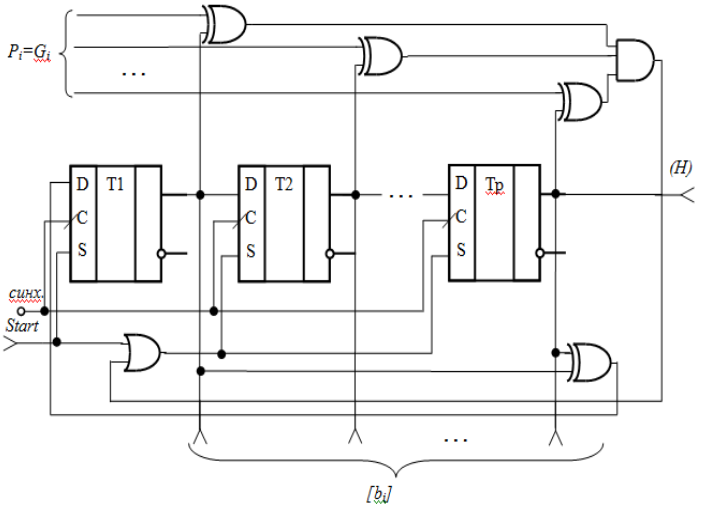
Продовження таблиці 4.5

1	2	3
Рademахера	+1, -1	+1
Галуа	-	+1, -1
Крестенсона	+1, -1	+1

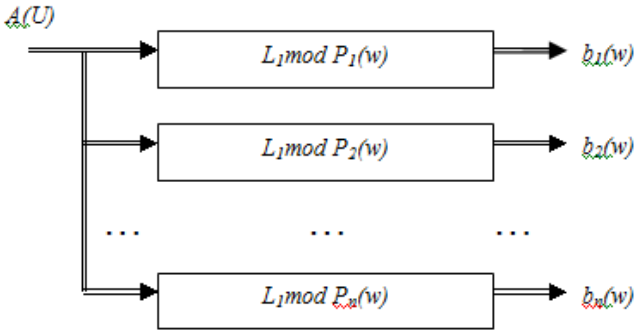
Таблиця 4.6 - Структури модульних лічильників у різних ТЧБ

ТЧБ	Структура
1	2
Унітарний	
Хаара	
Крейга	

Продовження таблиці 4.6

1	2
Рademахера	асинхронний +1  синхронний +1  асинхронний -1 
Галуа	

Продовження таблиці 4.6

Крестенсона	

Алгоритм даного класу пристроїв отримання залишків по модулю реалізується для ТЧБ унітарного, Хаара, Крейга та Радемахера згідно виразів:

$$b_i(U) = resA(U)(\mathrm{mod}\, P_i(U)),$$

де b_i – код залишку у відповідному ТЧБ;

A(U) – унітарний код числа з якого отримується залишок по модулю P_i;

P_i =(2, 3,…, U-1) – ціле число;

$b_i(H) = resA(U)(\mathrm{mod}\, P_i(H))$ –Хаара;

$b_i(K) = resA(U)(\mathrm{mod}\, P_i(R))$ – Крейга;

$b_i(R) = resA(U)(\mathrm{mod}\, P_i(R))$ – Радемахера.

У загальному випадку обчислення залишку та рангу числа А на основі модульного лічильника у ТЧБ (*w*) описується виразами:

$$A = a_i \cdot P_i + b_i;\ b_i(w) = resA(U)(\mathrm{mod}\, P_i(w));\ a_i(U) = \left(\frac{A - b_i}{P_i}\right)(U),$$

де $b_i(w)$ - паралельний код залишку числа А по модулю P_i у ТЧБ (*w*);

$a_i(U)$ – унітарний код рангу числа А по модулю P_i у ТЧБ (*w*).

Алгоритм обчислення залишку у ТЧБ Галуа виконується згідно виразу :

$$G_{i+1} = res(x_i \cdot G_i \oplus x_{i-1} \cdot G_{i-1} \oplus \ldots \oplus x_{i-n} \cdot G_{i-n}) \bmod 2,$$

де $x_i\ x_{i-1}\ x_{i-n}$ - незвідний поліном (ключ) поля Галуа $G\binom{n}{2}$.

При цьому, особливістю ТЧБ Галуа є те, що, на відміну від інших базисів, на паралельному виході лічильника Галуа формується паралельний бінарний код залишку, а на послідовному виході формується послідовний біт-

орієнтовний код Галуа залишку. Реалізація відповідного модульного лічильника виконується на базі синхронних D-тригерів згідно значенню модуля P_i заданого в коді Галуа, що забезпечує високу швидкодію даного класу процесорних модулів і меншу апаратну складність по відношенню до синхронних пристроїв визначення залишку у базисі Радемахера.

Алгоритм обчислення залишку на основі модульних лічильників базису Крестенсона реалізується згідно виразів:

$$b_i(C) = res(U)(\mathrm{mod}\, P_i(C)),\ \text{якщо}\ P_i = P_i^2(R);$$

$$C_k = \begin{cases} 1, & (b_i + b_j) = P_1 + 1, \\ 2, & (b_i + b_j) = P_2 + 2, \\ \ldots & \\ k, & (b_i + b_j) = P_1 + k, \\ \ldots & \\ P-1, & (b_i + b_j) = P + (P-1), \end{cases}$$

тобто

$$C(b_1(w), b_2(w), \ldots, b_j(w), \ldots, b_k(w)) = resA(U)\,\mathrm{mod}(P_1(w), P_2(w), \ldots, P_j(w), \ldots, P_k(w)),$$

якщо $P_i = \prod_{j=1}^{k} P_j$.

4.2.2 Аналого-цифровий перетворювач в базисі Крестенсона-Хаара та його характеристики

Принципова схема АЦП, який перетворює вхідні аналогові сигнали у коди ТЧБ Хаара-Крестенсона представлена модулями $p_i(2,3,5)$ на рис.4.4.

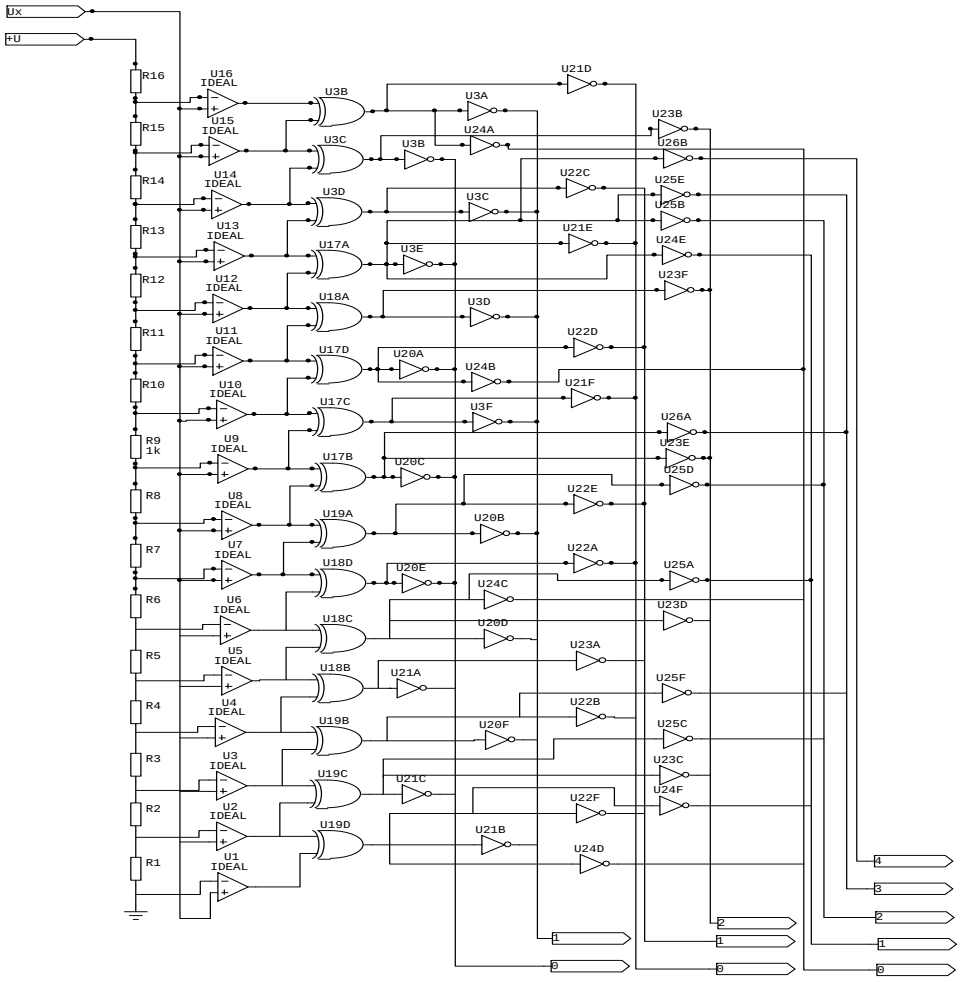


Рисунок 4.4 - Принципова схема АЦП базису Хаара-Крестенсона

Оцінка часової складності такого класу АЦП розраховується на основі виразу $\tau_{ACP} = \tau_k + \tau_{LE} + \tau_{VEN}$, де $\tau_k, \tau_{LE}, \tau_{VEN}$ відповідно часова складність компаратора, логічного елемента та вентиля. Згідно даних часової затримки сигналів в елементах приведених в [33] $\tau_k = 10\nu$, $\tau_{LE} = \nu$, $\tau_{VEN} = \nu$, тобто $\tau_{ACP} = 12\nu$.

Апаратна складність АЦП Крестенсона розраховується за формулою $A_{ACP} = 2^k \cdot (R + k + LE + \log_2 k / 2)$, де R – еталонний резистор, k – компаратор, *LE*– логічний елемент «виключаюче АБО».

4.2.3 Модульні шифратори Радемахера-Крестенсона

Структура модульних шифраторів Радемахера-Крестенсона запропонована в роботі [87] .

При 4-бітній розрядності компонентів процесора оптимальний набір модулів з максимальним діапазоном кодування задається масивом чисел:

$$p_i=(3, 5, 7, 8, 11, 13),$$

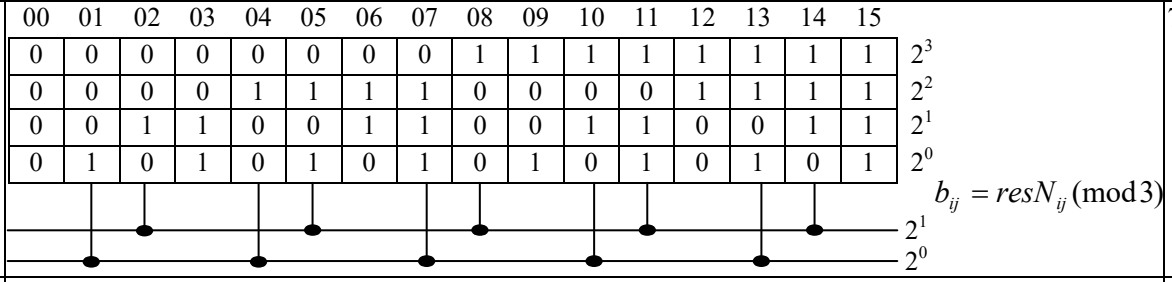
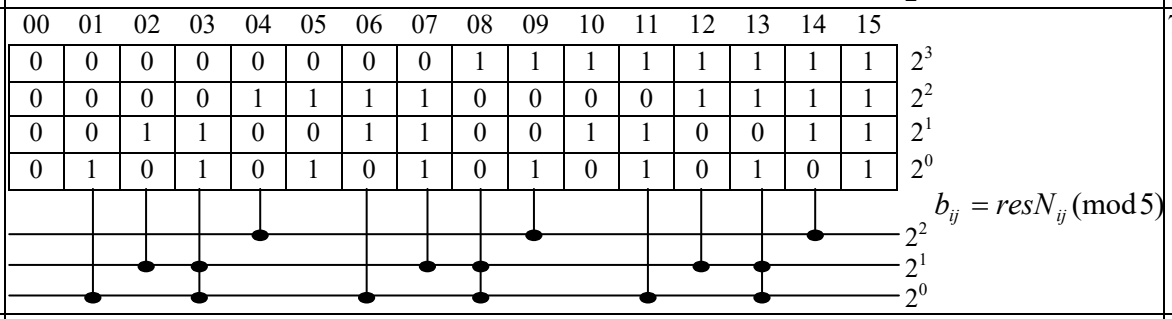
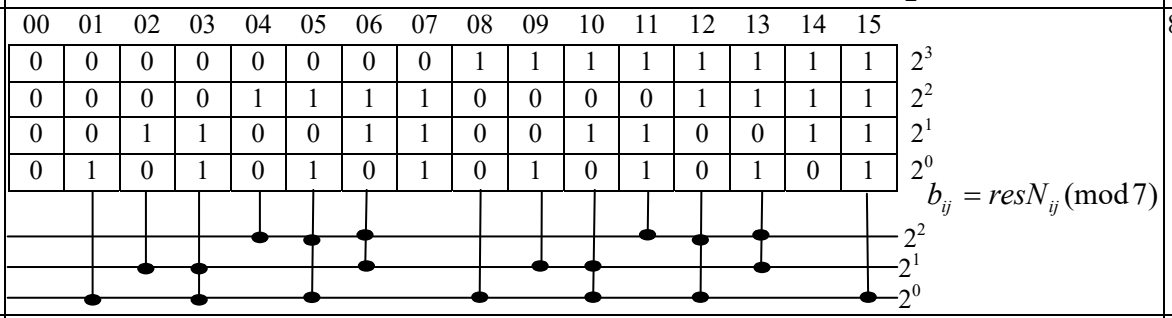
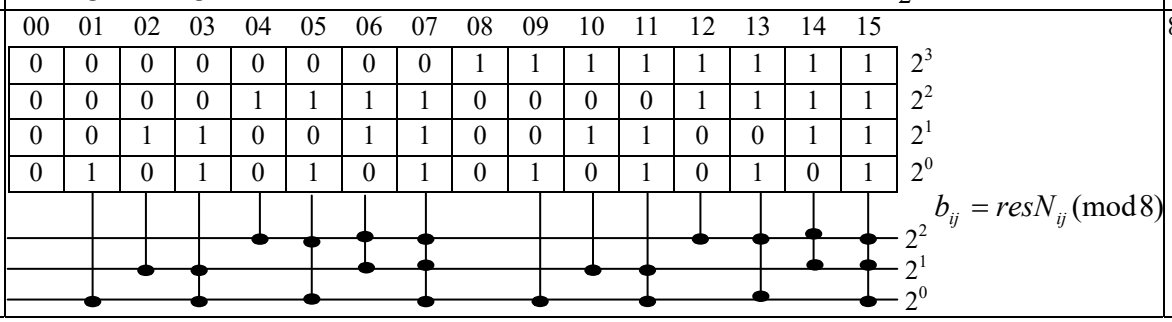
що відповідає діапазону кодування:

$$P=120120,$$

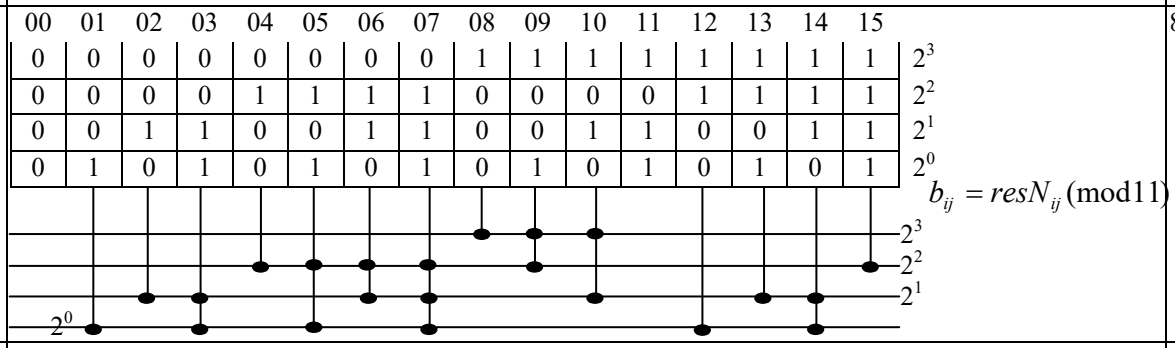
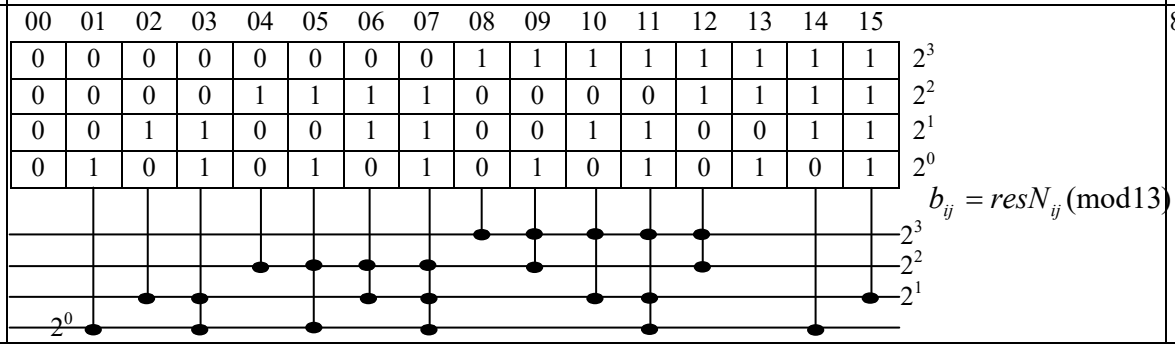
що достатньо для реалізації швидкодіючого 16-бітного процесора в РСЗК.

У табл.4.9 приведений приклад реалізації дешифраторів 4-бітних компонентів 16-бітного процесора залишкових класів.

Табл.4.9 – Дешифратори 4-бітних компонентів 16-бітного процесора залишкових класів

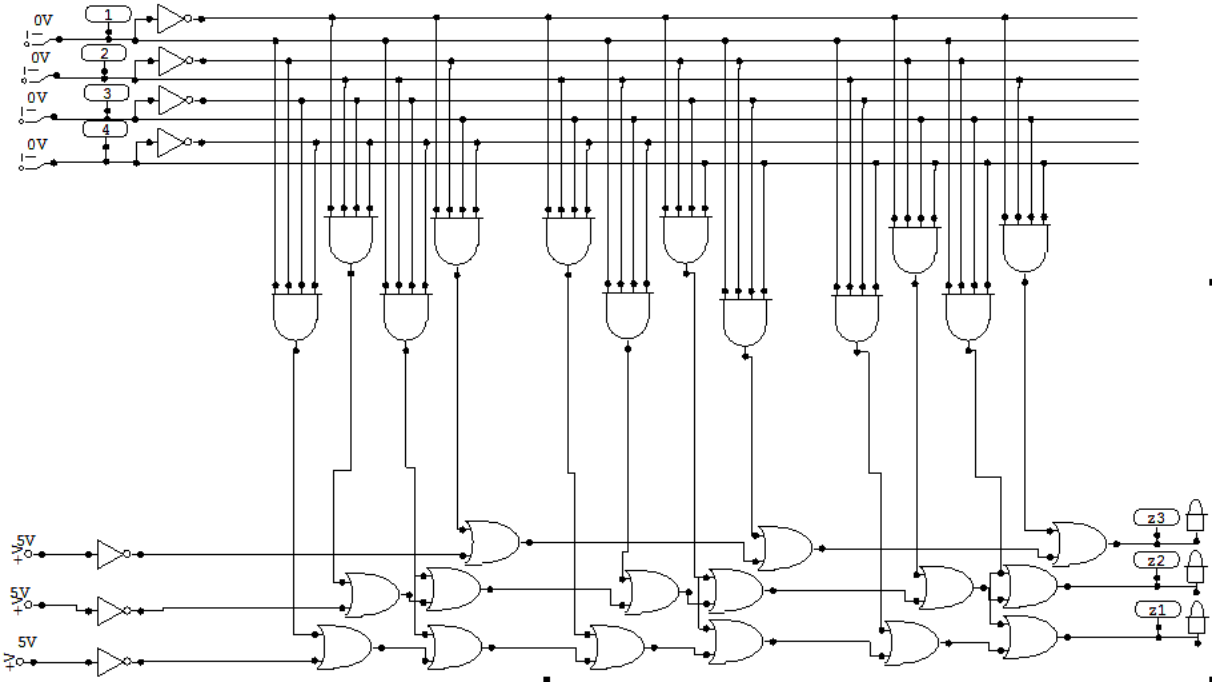
Модуль p_i		Число вентелів
3		74
5		79
7		83
8		88

Продовження таблиці 4.9

1	2	3
11		86
13		88

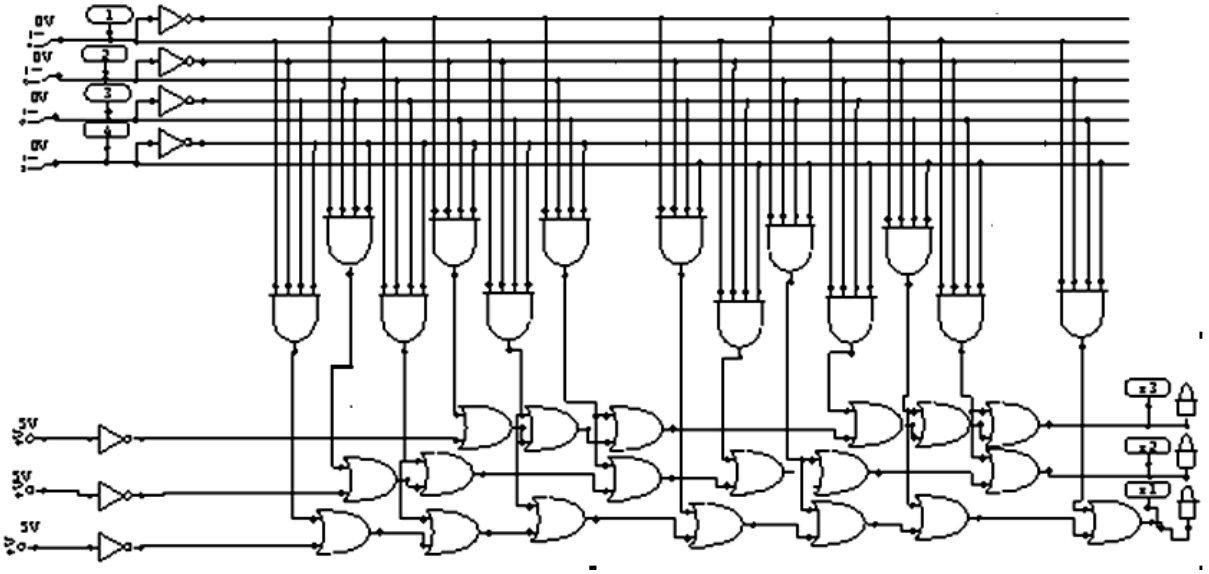
На рис.4.5 зображені приклади принципових схем обчислення залишку по модулях P=5, 7, 8.

Синтез, проектування та моделювання роботи таких модульних шифраторів виконаний в середовиці CircuitMaker в логічному базисі «І», «АБО», «НЕ».

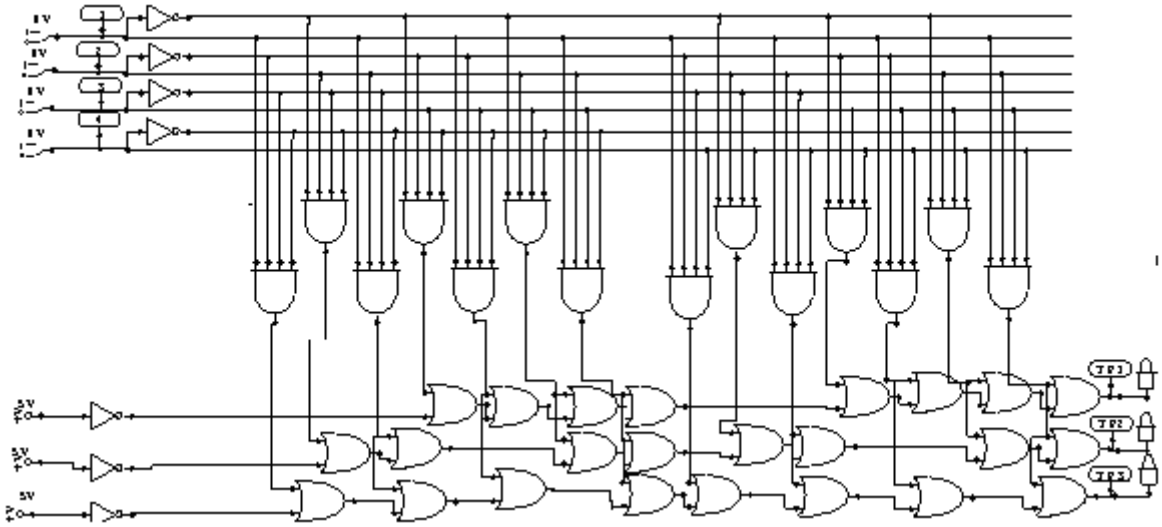


а)

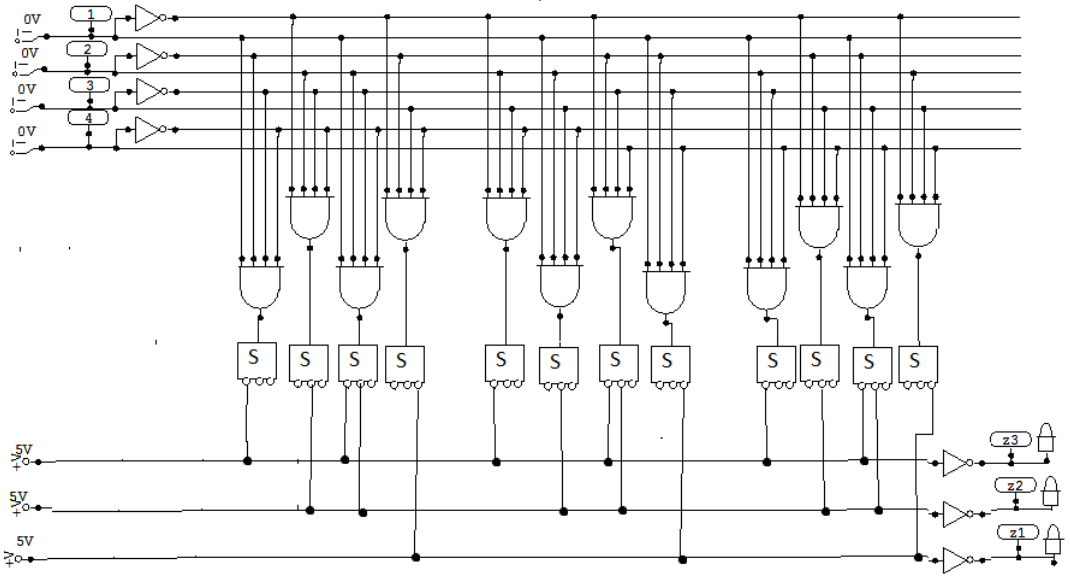
Рисунок 4.5 - Принципова схема обчислення залишку по модулях P=5(а), P=7(в), P=8(в), P=5(г) провідне АБО. S – селектор.



б)



в)



г)

Рисунок 4.5 - Продовження.

Часова складність шифраторів Радемахера-Крестенсона на основі структурної схеми з використанням послідовно включених елементів «АБО»,

пропорційна розрядності модуля p_i у базисі Хаара, тобто $\tau_{SH} = (p_i + 1)\nu$. При реалізації шифратора на основі схеми «провідне АБО» рис.4.6 часова складність шифратора суттєво зменшується $\tau_{SH} = 2\nu$. Відповідно, апаратна складність розглянутих схемотехнічних реалізацій шифраторів визначається згідно виразів $A_{SH} = 2^k \cdot (p_i + 1)$ та $A_{SH} = 2^k$ (рис.4.7).

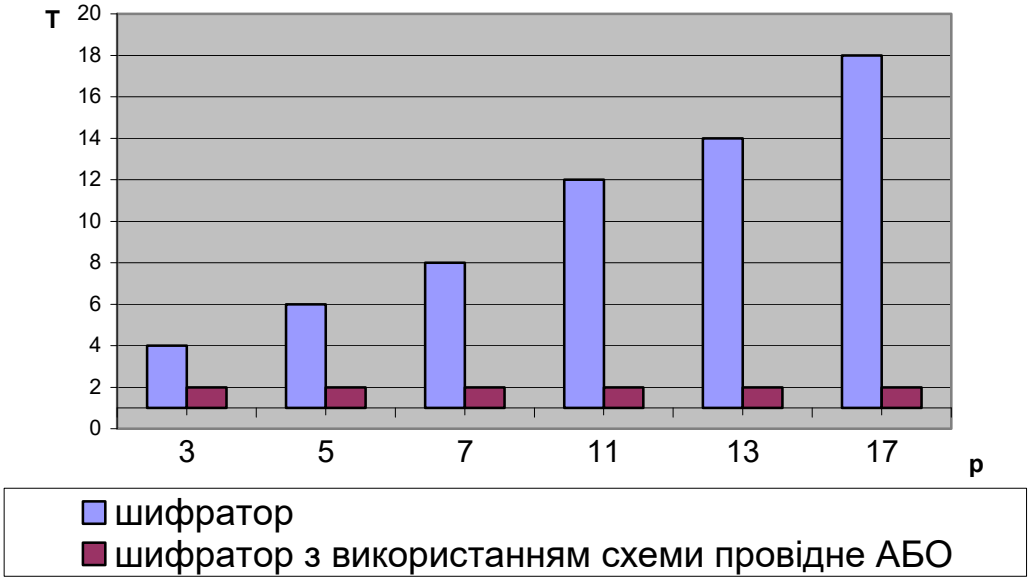


Рисунок 4.6 - Часова складність шифраторів Радемахера-Крестенсона

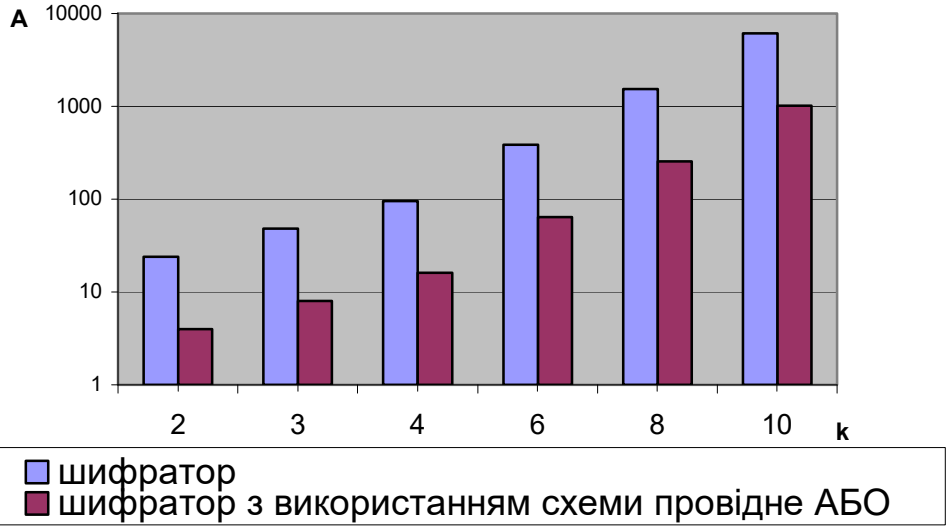


Рисунок 4.7 - Апаратна складність розглянутих схемотехнічних реалізацій шифраторів Радемахера-Крестенсона

З наведених діаграм часової та апаратної складностей (рис.4.6 та 4.7) видно, що при реалізації шифраторів доцільніше використовувати схеми «провідне АБО», які дозволяють зменшити апаратну складність в 6 разів.

4.3 Розробка міжбазисного перетворювача на основі способу з пам’яттю

Спосіб визначення залишку двійкового числа належить до систем перетворення інформації, які можуть бути використані для перетворення великорозрядних двійкових чисел по великорозрядних модулях у системах передавання та захисту інформації від несанкціонованого доступу, а також побудови спецпроцесорів в системі залишкових класів теоретико-числового базису Крестенсона.

Відомий спосіб визначення залишку двійкового числа, що ґрунтується на виконанні операції ділення двійкового числа на заданий модуль, і отримання найменшого невід’ємного залишку згідно алгоритму ділення по модулю двійкових чисел в доповнюючих кодах [9].

Недоліком такого способу є велика обчислювальна складність визначення залишку, оскільки його реалізація потребує багаторазового виконання операцій додавання доповнюючих кодів та операції порівняння чисел, що приводить до низької швидкодії обчислень, оскільки операція ділення потребує $(n+1-m)\cdot m$ - тактів виконання операцій сумування, $(n+1-m)$ - операцій порівняння та $(n-m)$ - операцій зсуву процесора при n-розрядності ділимого та m-розрядного дільника.

Найбільш близьким за суттю до розробленого міжбазисного перетворювача є спосіб отримання залишків двійкових чисел по заданому модулю, в якому *n*-розрядне двійкове число з вхідної шини записують в регістр пам’яті, а з вихідної шини знімають *m*-розрядний кінцевий код залишку цього числа по *m*-розрядному модулю, що ґрунтується на використанні розмежованої системи числення залишкових класів шляхом отримання сукупності залишків кожного одиничного розряду двійкового числа та їх лінійного сумування по

заданому модулю, який потребує n операцій сумування залишків для n-розрядного двійкового числа та наявності n- суматорів по модулю [101].

Проте такий спосіб характеризується великою часовою та апаратною складністю, оскільки для отримання залишку n-розрядного числа необхідне знаходження n окремих залишків для кожного окремого і-того розряду двійкового числа з наступним n-разовим їх сумуванням по модулю.

Розроблений спосіб визначення залишку двійкового числа здійснюється шляхом послідовного зчитування n-розрядного двійкового числа, починаючи зі старших розрядів, та послідовного отримання m-розрядних кодів залишків по модулю з постійної пам'яті, що дозволяє підвищити швидкодію отримання текучих та кінцевого залишку двійкового числа по модулю за n операцій зсуву та вибірки з пам'яті, в яких відсутні операції наскрізних переносів.

Поставлена задача вирішується завдяки тому, що спосіб визначення залишку двійкового числа по модулю, в якому n-розрядне двійкове число з вхідної шини записують в регістр пам'яті, а з вихідної шини знімають m-розрядний кінцевий код залишку цього числа по m-розрядному модулю, згідно з розробленим способом вводиться те, що двійковий код порозрядно зчитують, починаючи зі старших розрядів, сумують його з подвоєним кодом попереднього залишку, починаючи з його нульового, та формують повний код залишку по модулю з постійної пам'яті, який після n повторень таких операцій зчитується як кінцевий код залишку, починаючи зі старшого розряду.

Суть розробленого способу полягає у тому, що в способі отримання залишку, по відношенню до існуючих, відсутня операція порівняння, яка потребує використання суматорів, що, у свою чергу, приводить до виникнення наскрізних переносів, які знижують швидкодію визначення залишку. У запропонованому способі, замість вказаних операцій, здійснюється проста вибірка з постійної пам'яті з врахуванням модульної операції.

На рис.4.8 показана формалізована функціональна структура пристрою, який реалізує відомий спосіб визначення залишку двійкового числа [101, 111], де: 1 – вхідна шина двійкового числа X; 2 – n-розрядний регістр пам'яті; 3 – вхідна шина m-розрядного коду модуля P; 4 – дешифратори 2^i значень бітів

числа X в коди залишків b_i по модулю P; 5 – n-розрядні суматори кодів залишків b_i та b_j по модулю P; 6 – вихідна шина коду залишку b_0.

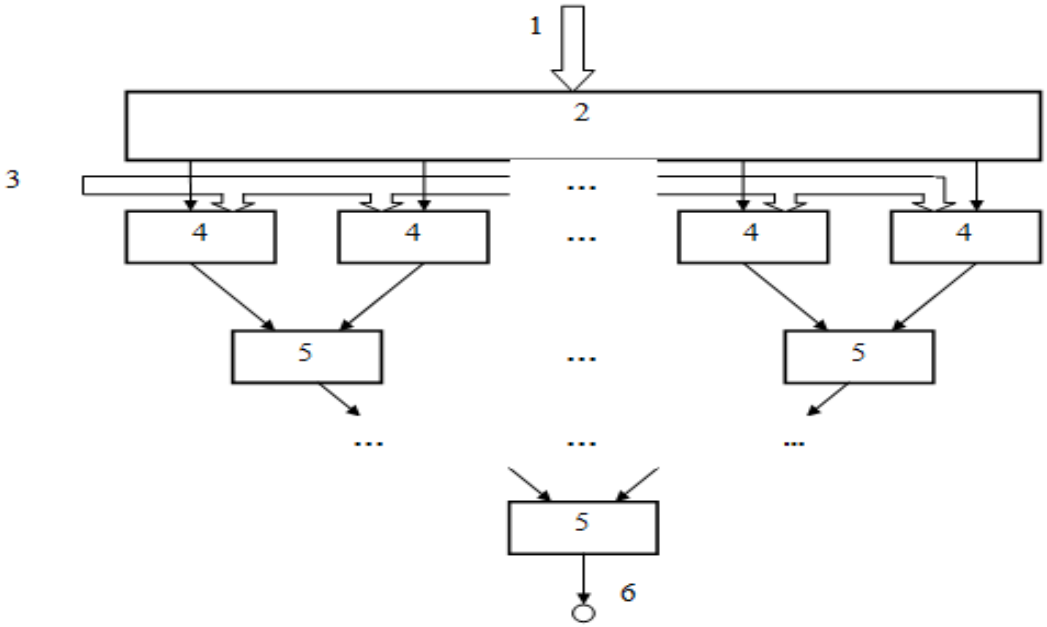


Рисунок 4.8 - Структура пристрою визначення залишку двійкового числа

На рис.4.9 зображена функціональна структура пристрою, який реалізує запропонований спосіб визначення залишку двійкового числа по модулю P, що складається: 1 – вхідна шина двійкового числа X та залишку $b_0=0$; 2 – $n+m$-розрядний регістр пам'яті та зсуву; 3 – вхідна шина m-розрядного коду модуля P; 4 – постійний запам'ятовуючий пристрій; 5 – вихідна шина проміжних та кінцевого залишку b_0 по модулю P.

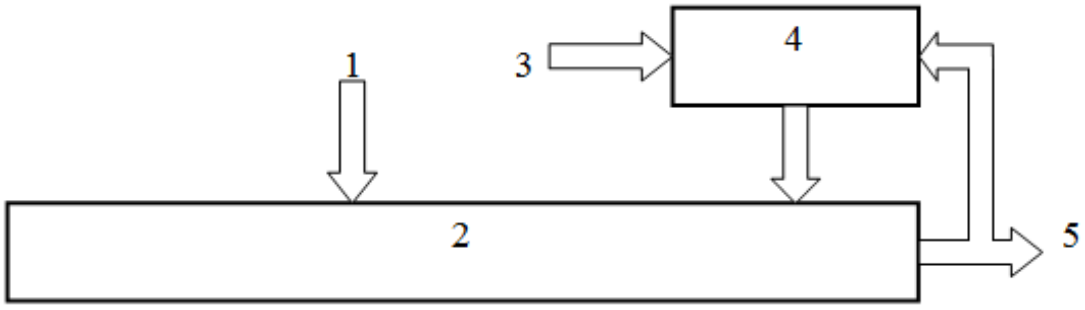


Рисунок 4.9 - Структура пристрою визначення залишку двійкового числа з використанням ПЗП

У регістр пам'яті та зсуву – 2 з вхідної шини – 1 заноситься n-розрядний код числа X, молодші розряди якого записуються у відповідні розряди регістра

2, починаючи зліва, а в інші *m*-розряди записуються нулі. Одночасно з вхідної шини – 3 в постійний запам'ятовуючий пристрій – 4 подається *m*-розрядний двійковий код модуля *P*. Після *n* зсувів з вихідної шини – 5 зчитують код кінцевого залишку b_0 числа *X* по модулю *P*, починаючи зі старшого розряду справа.

Для реалізації запропонованого способу регістр пам'яті та зсуву – 2 виконується відомою мікроелектронною схемою на D-тригерах з мультиплексорами на D входах, а постійний запам'ятовуючий пристрій на основі групи k *m*-адресних кристалів флеш-пам'яті.

Схемотехнічне представлення компонента отримання залишку приведене на рис.4.10

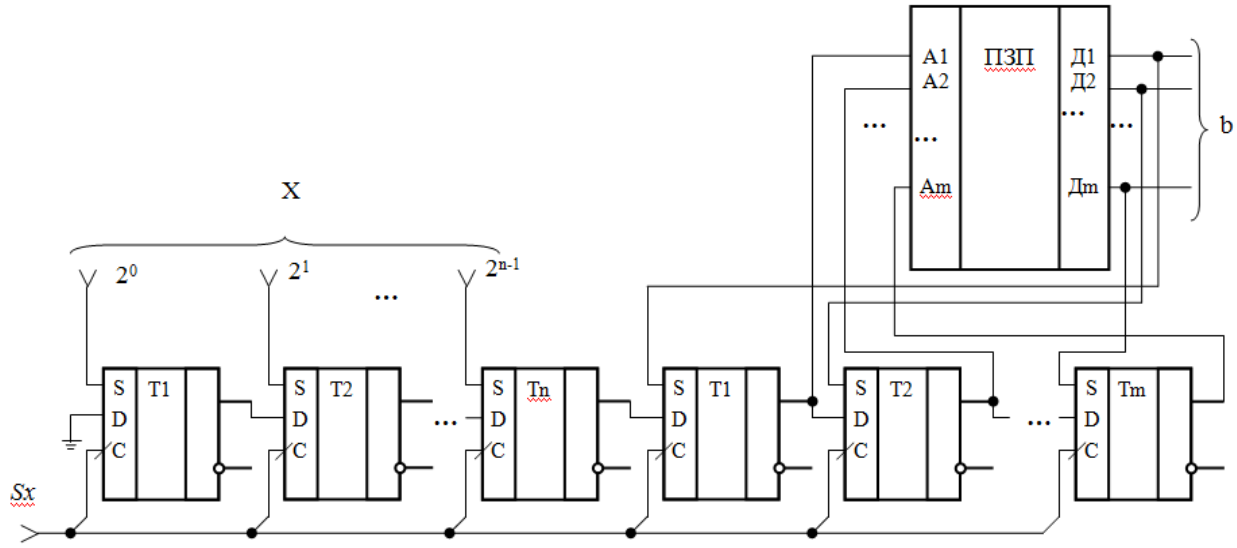


Рисунок 4.10 - Схемотехнічна реалізація пристрою визначення залишку двійкового числа з використанням ПЗП

Реалізація міжбазисного перетворювача на основі пристрою визначення залишку двійкового числа з використанням ПЗП буде мати структуру, показану на рис.4.11.

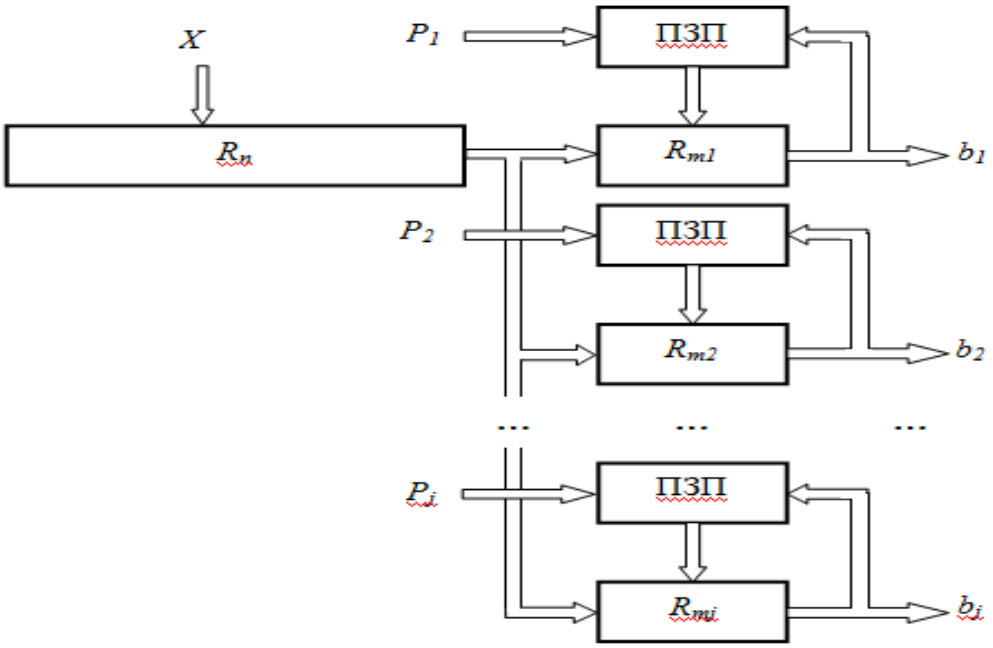


Рисунок 4.11 - Структура міжбазисного перетворювача Радемахера-Крестенсона на основі пристрою визначення залишку двійкового числа з використанням ПЗП

Для реалізації даного міжбазисного перетворювача використовуються відповідні набори модулів представлених в табл.4.8, що дозволяє продемонструвати переваги вибору модулів при різних розрядностях спецпроцесора.

$$A = n \cdot 2ЛЕ + \mathrm{k} \cdot 2ЛЕ + A_{ПЗП}$$

$$A_{ПЗП} = 2\mathrm{p} \cdot \mathrm{k}$$

$$\mathrm{k} = \hat{E}[\log_2 p]$$

Таблиця 4.8 - Набори модулів та їх характеристики для різної розрядності спецпроцесора

Характеристики модулів	Розрядність спецпроцесора (біт)					
	8	16	32	64	128	256
Розрядність модулів min (біт)	3	4	5	6	7	8
Кількість модулів min	3	5	7	12	20	34
Розрядність модулів max (біт)	5	6	7	8	9	9
Кількість модулів max	2	3	5	9	15	31

Апаратна складність такого міжбазисного перетворювача згідно табл.4.8 наведена на рис.4.12.

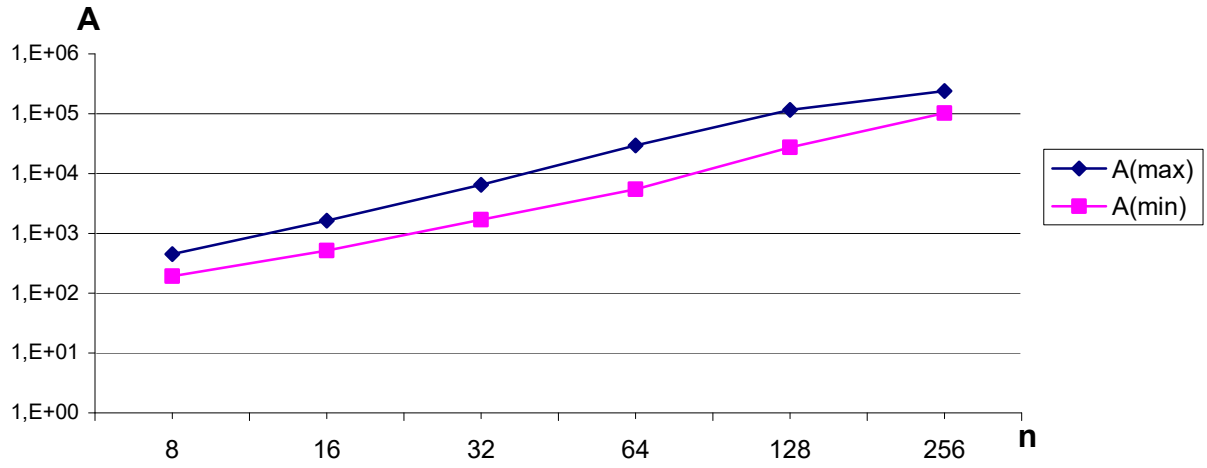


Рисунок 4.12 - Апаратна складність міжбазисного перетворювача з використанням ПЗП з різними наборами модулів

З графіка видно (див.рис.4.12), що апаратна складність даного міжбазисного перетворювача менша при наборі модулів min в 2, рази ніж при при наборі модулів з розрядністю max, тому для реалізації даного спецпроцесора доцільніше застосовувати набори модулів (додаток Б) з меншою розрядністю (рис.4.13).

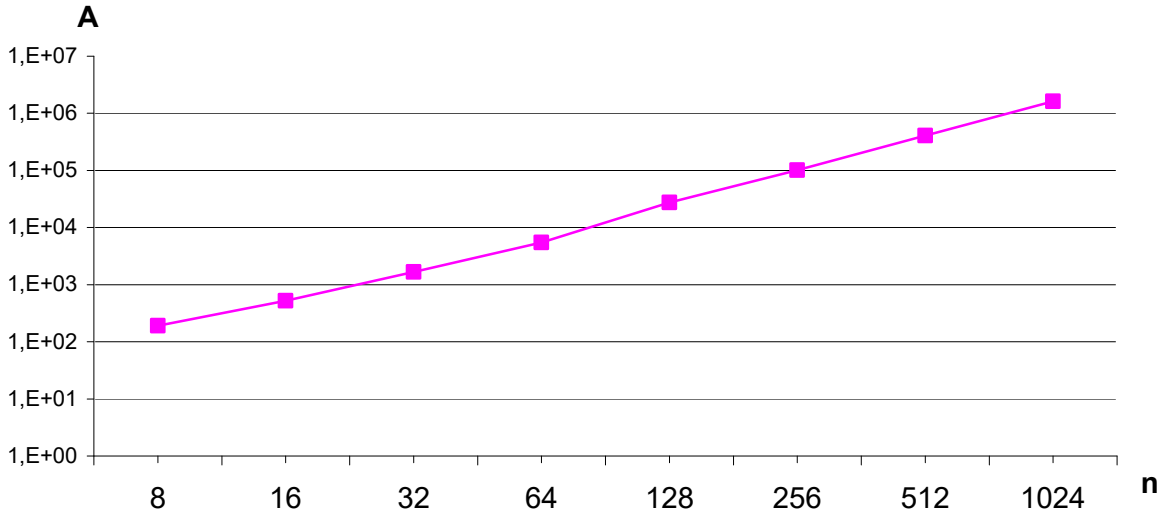


Рисунок 4.13 - Апаратна складність міжбазисного перетворювача з використанням ПЗП

Слід зазначити, що часова складність відомого алгоритму [101, 111] буде обчислюватись згідно наступного співвідношення:

$$O2(n) = mn^2 + mn + m^2 n + 2n - 2m + 1.$$

Скориставшись пірамідальним алгоритмом для обчислення залишку [101, 111] отримаєм наступний вираз часової складності: $O3(n) = n\log^2(n)$.

Часова складність розробленого методу обчислення залишку великорозрядних чисел по заданому модулю, згідно рекурентного співвідношення $b_i = (a_i + 2b_i - 1) \bmod p_i$, обчислюється згідно виразу: $O1(n) = 2n$. Результатом чисельного експерименту показано, що розроблений метод характеризується меншою часовою складністю на 2-3 порядки в порівнянні з відомими (рис.4.14).

Отже, враховуючи проведені дослідження, розроблений метод на основі використання рекурентного співвідношення доцільно використовувати в міжбазисних перетвореннях.

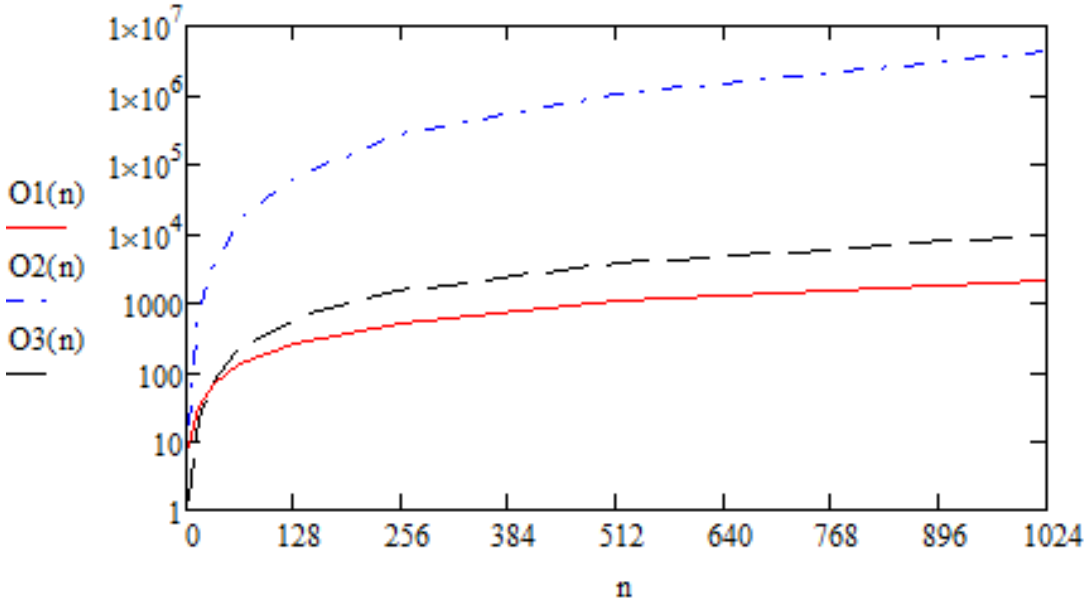


Рисунок 4.14 – Часова складність обчислення залишків по модулю.

Наприклад, для використовуваних сучасних способів захисту інформації RSA та Ель-Гамаля з розрядністю модулів 512-1024 біти потрібно 4 кристали флеш-пам’яті 16 ГБ. При тактовій частоті регістра пам’яті та зсуву – 2 100 МГц

затримка часу визначення одного залишку 1024-х бітного числа X складає близько 1 млс.

Операція обчислення залишку розробленим способом потребує n тактів в регістрі пам'яті та зсуву – 2 та n тактів вибірки коду залишку з постійного запам'ятовуючого пристрою – 4, тобто рівне $2n$, у випадку, якщо двійкове число буде займати 512 біт, а модуль, за яким обчислюють залишок, буде від 2 до 128 біт то швидкодія, в порівнянні з відомим способом, зросте від 2-х до 48-и разів.

4.4 Реалізація швидкодіючого перетворювача Радемахера-Крестенсона на основі рандомізаторів

Розроблений перетворювач Радемахера-Крестенсона відноситься до обчислювальної техніки і може бути використаний в якості міжбазисного перетворювача обчислювальних пристроїв, які працюють у двійковій системі числення базису Радемахера, в систему числення залишкових класів теоретико-числового базису Крестенсона.

Відомий пристрій для перетворення десяткового коду в систему залишкових класів [102], який містить вхідні регістри для запису вхідного десяткового числа, розрядні перетворювачі по кожному модулю системи, вхідні шини для подачі розрядів числа та виходи кодів залишків b_i по модулю P_j системи залишкових класів.

Недоліком такого пристрою є низька швидкодія та висока апаратна складність, які обумовлені тим, що він містить арифметичні пристрої відповідного числа модулів системи на цифрових елементах розподілу струму.

Також, відомий пристрій для перетворення чисел з десяткової системи числення в систему залишкових класів [103], який містить перетворювачі степенів основи в системі залишкових класів, блоки множення і сумування по модулю з'єднані між собою відповідним чином.

Недоліком відомого пристрою є низька швидкодія та висока апаратна складність, обумовлена наявністю великої кількості блоків множення та сумування по кожному P_j – модулю в системі залишкових класів.

Відомий прототип до запропонованого перетворювача Радемахера-Крестенсона є пристрій для перетворення чисел з позиційної системи числення в систему залишкових класів [104], який містить шини для подачі K-розрядного позиційного числа, перетворювачі степенів розрядів числа в позиційній системі числення по модулю P_j, K виходів кодів залишків системи залишкових класів, блоки множення та сумування по модулю P_j, які з'єднані відповідним чином.

Недоліком пристрою перетворення чисел з позиційної системи числення в систему залишкових класів є низька швидкодія та висока апаратна складність, обумовлена наявністю блоків множення та сумування по модулю P_j.

Поставлена задача підвищення швидкодії та зменшення апаратної складності вирішується завдяки тому, що пристрій для перетворення чисел з позиційної системи числення в систему залишкових класів містить шини для подачі K –розрядного позиційного числа, перетворювачі степенів розрядів числа в позиційній системі числення по модулю P_j, виходи кодів залишків системи залишкових класів, згідно з розробленим міжбазисним перетворювачем, додатково введені K комутаційних мультиплексорів по кожному модулю системи залишкових класів P_j, перші входи яких підключені до відповідних шин вхідного K –розрядного двійкового числа, а другі входи i-тих комутаційних мультиплексорів підключені до виходів i-1-ших комутаційних мультиплексорів починаючи, з K-2 до нульового розряду, при чому, на одиничний вхід K-1-го комутаційного мультиплексора подається старший біт двійкового числа базису Радемахера, а виходи нульового комутаційного мультиплексора є виходами чисел залишків b_i по модулю P_j системи залишкових класів.

Міжбазисний перетворювач Радемахера-Крестенсона ілюструється зображеною структурною схемою пристрою (рис.4.14) [111, 112]: 1 – вхідні шини K –розрядного позиційного числа, 2 – комутаційні мультиплексори, 3 – виходи коду b_i системи залишкових класів.

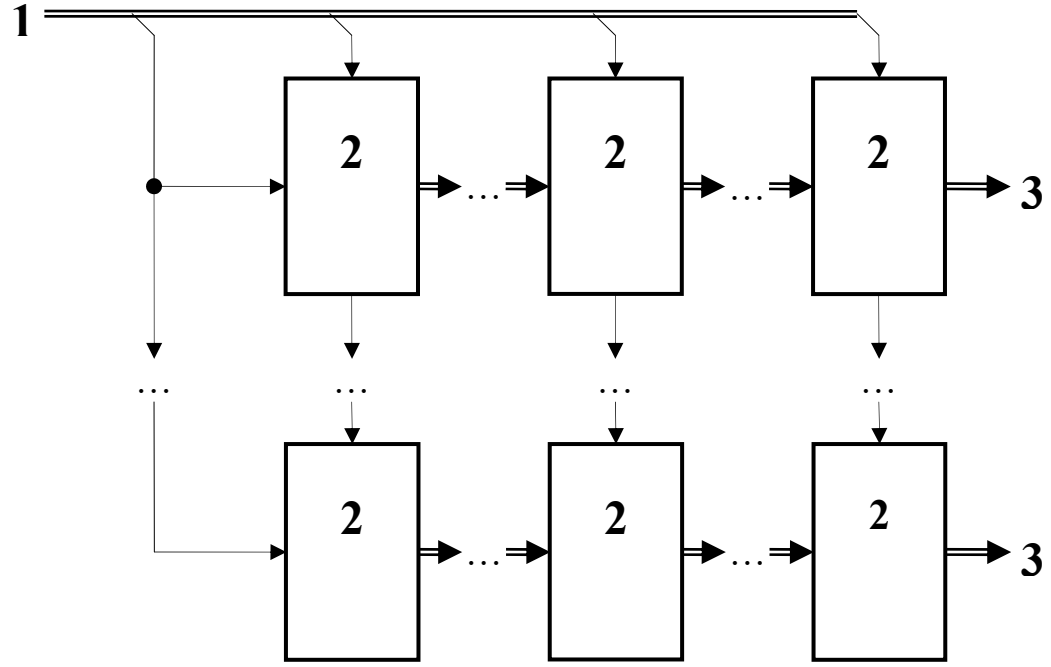


Рисунок 4.14 - Структурна схема перетворювача Радемахера-Крестенсона на основі рандомізаторів

На рис.4.15 зображена структурна схема комутаційного мультиплексора: 2.1– рандомізатор по модулю P_j, 2.2 – інкрементний пристрій по модулю P_j, 2.3 – P-канальний двохвходовий мультиплексор.

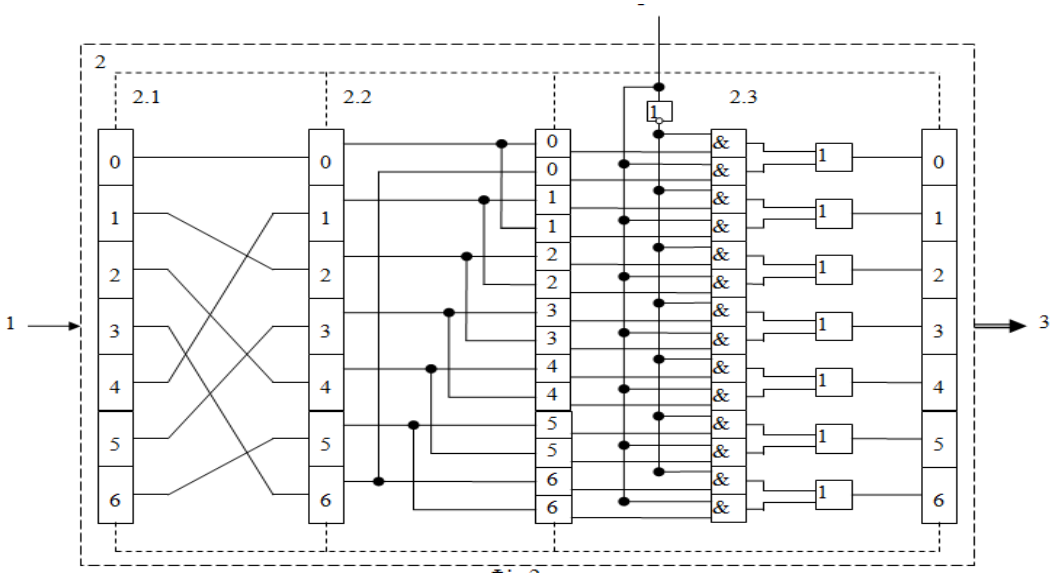


Рисунок 4.15 - Структура комутаційного мультиплексора

На вхідну шину – 1 подається код К-розрядного двійкового числа базису Радемахера, при цьому на перші входи *i-1*-тих комутаційних пристроїв всіх *j*-

тих розрядів вихідних кодів системи залишкових класів подаються двійкові значення 0 або 1 X_{i-1}-тих вхідних шин. При подачі на одиничні входи всіх j-тих К-1-ших комутаційних мультиплексорів – 2 сигналу старшого біту X_i відбувається однозначне формування на виходах всіх X_j нульового розряду вхідного числа унітарних кодів Хаара залишків b_j системи залишкових класів. У результаті за один такт переключення мультиплексорів 2 відбувається перетворення K-розрядних чисел базису Радемахера у систему залишкових класів базису Крестенсона у системі взаємопростих модулів $p_1, p_2, \ldots p_j, \ldots p_m$, де m – число взаємопростих модулів.

Оскільки рандомізатори по модулю p_j організовані на основі провідникової схеми без активних елементів то швидкодія пристрою визначається часом паралельного переключення мультиплексорів, тобто за 1 такт. А також, пристрій характеризується регулярністю структури і легко проектується на ПЛМ, що забезпечує зменшення апаратної складності порівняно з відомим пристроєм, та надійністю мікроелектронної реалізації.

Оцінку часової та апаратної складності спецпроцесора міжбазисного перетворення на основі рандомізаторів та мультиплексорів розраховуємо згідно виразів: $\tau = 2LE$, що відповідає тривалості переключення двох послідовно підключених елементів мультиплексора. Оскільки всі мультиплексори переключаються одночасно при подачі на їх входи бітових значень великорозрядного двійкового числа, то швидкодія такого міжбазисного перетворювача не залежить від перетворюваного числа базису Радемахера.

Оцінка апаратної складності розробленого міжбазисного перетворювача розраховується згідно виразу $A = 2p + p + 1 + (9p \cdot 0.001)$, де 0,001 – апаратна складність з’єднань міжбазисного перетворювача.

Апаратна складність такого міжбазисного перетворювача згідно наборів модулів табл.4.8 наведена на рис.4.16 та рис.4.17.

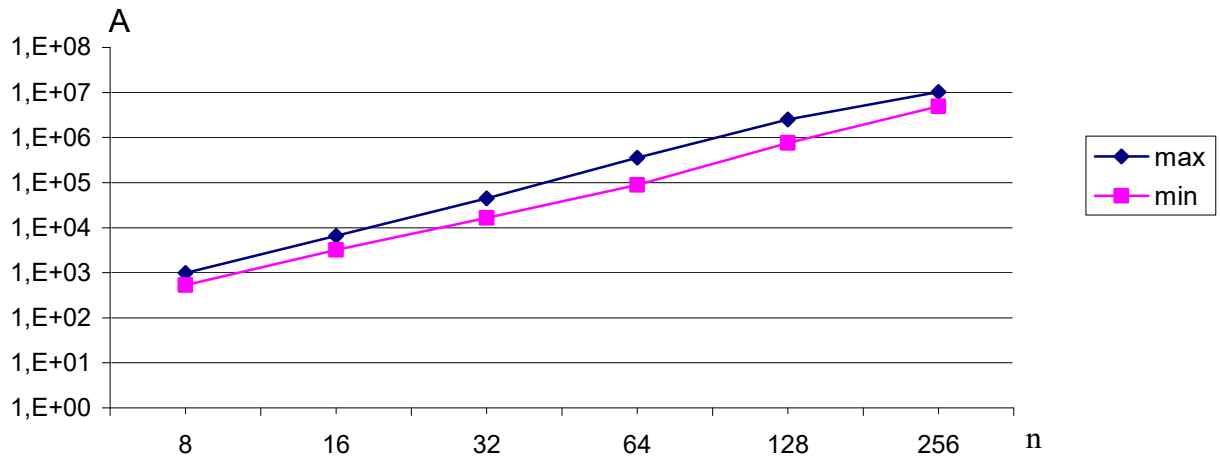


Рисунок 4.16 - Апаратна складність міжбазисного перетворювача на основі рандомізаторів з різними наборами модулів

З графіка видно (див.рис.4.16), що апаратна складність даного міжбазисного перетворювача менша при наборі модулів min в 2 рази, ніж при при наборі модулів з розрядністю max, тому для реалізації даного спецпроцесора доцільніше застосовувати набори модулів (додаток В) з меншою розрядністю (рис.4.17).

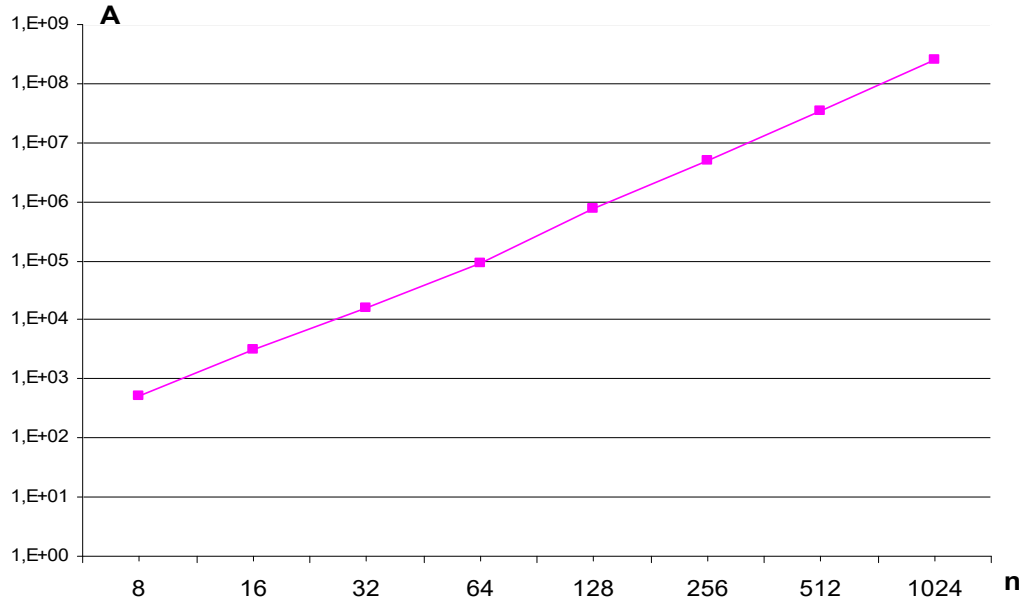


Рисунок 4.17 - Апаратна складність міжбазисного перетворювача на основі рандомізаторів

4.5 Розрахунок системи взаємопростих модулів високопродуктивного кореляційного спецпроцесора в базисі Хаара-Крестенсона

Функціональна структура спецпроцесора обчислення коваріаційної функції на основі числення системи залишкових класів теоретико-числового базису Крестенсона та представлення кодів залишків системи взаємопростих модулів у базисі Хаара приведена на рис.4.18.

Обчислення коваріаційної функції здійснюється згідно виразу:

$$K_{xx}(j) = \frac{1}{N}\sum_{i=0}^{N-1} x_i \cdot x_{i-j},$$

де N – об'єм вибірки, x_i - цифрові відліки;

$j \in \overline{0,m}$ - часовий дискретний зсув;

m – число точок коваріаційної функції.

З метою розпаралелення та підвищення швидкодії виконання обчислювальних операцій у базисі Хаара-Крестенсона на виході аналого-цифрового перетворювача паралельно формуються коди базису Хаара системи взаємопростих модулів перетворення Крестенсона:

p_1	p_2	…	p_i	...	p_k
$b_i(H)$=(010…0)	(101…0)	…	(011…1)	…	(100…1),

де $p_1, p_2, \ldots, P_I, \ldots, p_k$ – взаємно прості модулі системи залишкових класів;

$b_i(H)$ –паралельні коди базису Хаара.

На рис. 4.18 позначені наступні модулі: АЦП – паралельний АЦП, який перетворює вхідні аналогові сигнали X(t) у масив кодів Хаара, які відповідають залишкам системі взаємно простих модулів базису Крестенсона; МК(+), МК(×) – відповідно К матриць модульного сумування та множення у базисі Хаара; ПЗП – постійний запам'ятовуючий пристрій, який перетворює коди Хаара-Крестенсона у вихідні коди значень коваріаційної функції базису Радемахера [112].

Таким чином, у процесі обчислення коваріаційної функції у кожному каналі процесора виконується матрично-модульне перемноження поточних та зміщених в регістрі відліків, а також накоплююче матрично-модульне

сумування кодів Хаара. У результаті у кожному j-му каналі по P_i модулю виконується операція модульного множення і на виходах формуються коди залишків згідно виразу:

$$b_m(H) = res[b_i(H) + b_{i-j}(H)] \bmod P_i . \qquad (4.1)$$

Структурна схема формування прямих кодів залишків у базисі Хаара в результаті модульного сумування згідно виразу (4.1) показана на рис.4.19. На пересіченні кодових шин матриці Хаара розміщений логічний елемент «І-НЕ», при чому виходи елементів, які відповідають одному значенню коду Хаара з'єднані між собою і утворюють вихідну шину зворотних паралельних кодів Хаара, які після інвекторів перетворюються в прямі коди базису Хаара, які представляють результати модульного сумування.

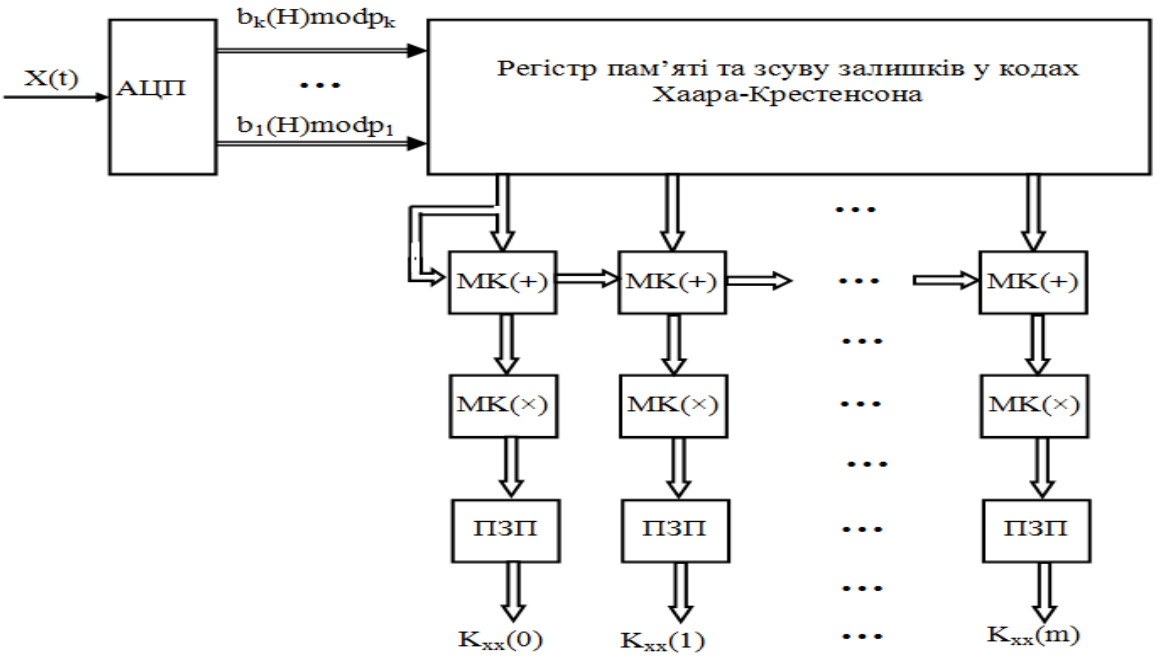


Рисунок 4.18 - Функціональна структура спецпроцесора обчислення коваріаційної функції у базисі Хаара-Крестенсона.

Отже, часова затримка виконання модульних операцій у матрицях модульного сумування та множення виконується за два мікротакти незалежно від розрядності АЦП та числа точок обчислювальної автоковаріаційної функції $\tau_m = 2\nu$, де ν - тривалість переключення мікроелектронного обладнання.

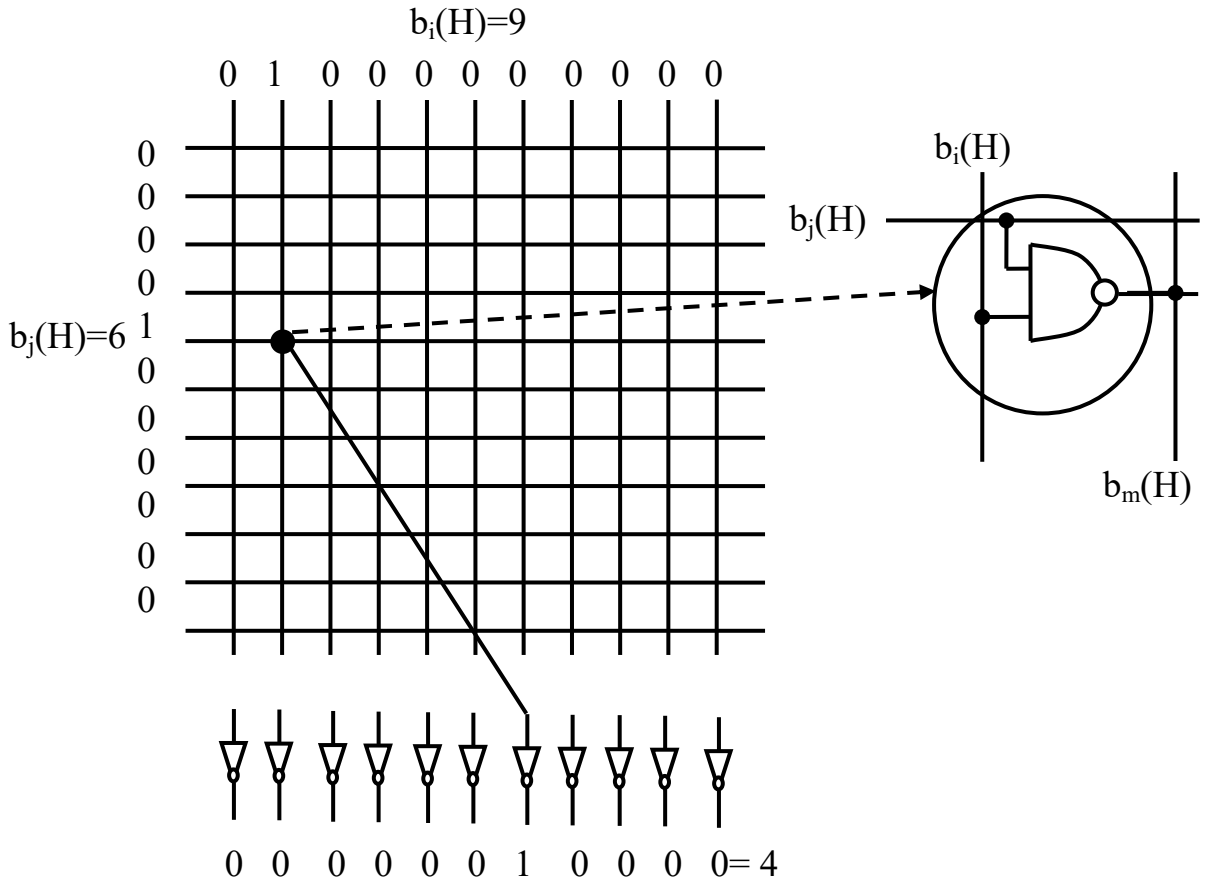


Рисунок 4.19 - Структура модульної матриці сумування кодів Хаара.

Аналогічну структуру має матриця модульного множення кодів Хаара.

Приклади розрахунку параметрів автокорелятора при різній розрядності АЦП K, числа каналів автокорелятора M і об’ємі вибірки N:

Приклад 1.

- вихідні дані автокорелятора K=4, M=16, N=256. Шукаємо набір взаємо простих модулів (P_1, P_2, …, P_k), добуток яких найменше перевищує суму добутків максимальних значень $x_i \cdot x_{i-j}$, тобто $\prod_{i=1}^{k} P_i \geq \sum_{i=1}^{256} x_i \max \cdot x_{i-j} \max$. Для нашого прикладу $x_i \max = 2^4 - 1 = 15$, отже $\sum_{i=1}^{256} 15^2 = 57600$.

Вибираємо набір наступних взаємнопростих модулів: p_1=11, p_2=13, p_3=15, p_4=31. Їх добуток рівний P=11·13·15·31=66495, що задовольняє діапазон кодування чисел в системі залишкових класів.

Приклад. 2

- вихідні дані автокорелятора K=8, M=16, N=256. Діапазон кодування чисел в кожному каналі автокорелятора $255^2 \cdot 256^2 = 16646400$.

Вибраний набір модулів: $p_1=25$, $p_2=27$, $p_3=28$, $p_4=29$, $p_5=31$. Їх добуток рівний $P=25 \cdot 27 \cdot 28 \cdot 29 \cdot 31 = 16991100$, що задовольняє умову однозначного кодування результатів обчислень у автокорелятері в системі залишкових класів базису Крестенсона.

У загальному випадку швидкодія спецпроцесора обчислення автоковаріаційної функції у базисі Хаара-Крестенсона з вихідними кодами у базисі Радемахера визначається згідно виразу:

$$\tau_n = \tau(АЦП) + \tau(R) + \tau(M+) + \tau(M\times) + \tau(ПЗП) \ ,$$

де $\tau(АЦП)$ згідно структури, показаної на рис.4.4, з врахуванням сумарного часу переключення послідовно включених компаратора, логічного елемента та мікроелектронного вентиля не перевищує 6ν, $\tau(R)$ регістра пам'яті та зсуву, який реалізується на D-тригерах рівний 2ν, а $\tau(ПЗП)$, яке реалізується на основі вентильних матриць не перевищує 2ν. У результаті отримаємо загальну оцінку швидкодії розробленого кореляційного спецпроцесора:

$$\tau_n = 6\nu + 2\nu + 2\nu + 2\nu = 12\nu \, .$$

Таким чином, при тактовій частоті роботи кристала мікроелектроної реалізації процесора 100 МГц час затримки обчислення автоковаріаційної функції не перевищує 120 нс.

При створенні великорозрядних процесорів у базисі Крестенсона у системі взаємно простих модулів можуть використовуватися наступні значення модулів. Один модуль з значенням 2^k, де k- ціле число; по одному модулю p^k, де p – просте число; модулі, рівні простим числам; складені модулі добутків простих чисел.

В таблиці 4.9 розраховані значення k – розрядних модулів для процесорів з різною розрядністю розробленою програмою (Додаток И).

Таблиця 4.9 – Набори модулів для процесорів з різною розрядністю

3 бітні модулі:

8 (Складене число: 2 ^ 3)	5 (Просте число)
7 (Складене число: 7 * 1)	3 (Просте число)

4 бітні модулі:

16 (Складене число: 2 ^ 4)	11 (Просте число)
15 (Складене число: 3 * 5)	7 (Просте число)
13 (Просте число)	

5 бітні модулі:

32 (Складене число: 2 ^ 5)	19 (Просте число)
31 (Складене число: 31 * 1)	17 (Просте число)
29 (Просте число)	13 (Просте число)
27 (Складене число: 3 ^ 3)	11 (Просте число)
25 (Складене число: 5 * 5)	7 (Просте число)
23 (Просте число)	

6 бітні модулі:

64 (Складене число: 2 ^ 6)	41 (Просте число)
63 (Складене число: 3 * 3 * 7)	37 (Просте число)
61 (Просте число)	31 (Просте число)
59 (Просте число)	29 (Просте число)
55 (Складене число: 11 * 5)	23 (Просте число)
53 (Просте число)	19 (Просте число)
47 (Просте число)	17 (Просте число)
43 (Просте число)	13 (Просте число)

7 бітні модулі:

128 (Складене число: 2 ^ 7)	73 (Просте число)
127 (Складене число: 127 * 1)	71 (Просте число)
125 (Складене число: 5 ^ 3)	67 (Просте число)
123 (Складене число: 41 * 3)	61 (Просте число)
121 (Складене число: 11 * 11)	59 (Просте число)
119 (Складене число: 17 * 7)	53 (Просте число)
113 (Просте число)	47 (Просте число)
109 (Просте число)	43 (Просте число)
107 (Просте число)	37 (Просте число)
103 (Просте число)	31 (Просте число)
101 (Просте число)	29 (Просте число)
97 (Просте число)	23 (Просте число)
89 (Просте число)	19 (Просте число)
83 (Просте число)	13 (Просте число)
79 (Просте число)	

Продовження таблиці 4.9

8 бітні модулі:

256 (Складене число: 2 ^ 8)	139 (Просте число)
255 (Складене число: 3 * 5 * 17)	137 (Просте число)
253 (Складене число: 23 * 11)	131 (Просте число)
251 (Просте число)	127 (Просте число)
247 (Складене число: 19 * 13)	113 (Просте число)
241 (Просте число)	109 (Просте число)
239 (Просте число)	107 (Просте число)
233 (Просте число)	103 (Просте число)
229 (Просте число)	101 (Просте число)
227 (Просте число)	97 (Просте число)
223 (Просте число)	89 (Просте число)
217 (Складене число: 31 * 7)	83 (Просте число)
211 (Просте число)	79 (Просте число)
199 (Просте число)	73 (Просте число)
197 (Просте число)	71 (Просте число)
193 (Просте число)	67 (Просте число)
191 (Просте число)	61 (Просте число)
181 (Просте число)	59 (Просте число)
179 (Просте число)	53 (Просте число)
173 (Просте число)	47 (Просте число)
167 (Просте число)	43 (Просте число)
163 (Просте число)	41 (Просте число)
157 (Просте число)	37 (Просте число)
151 (Просте число)	29 (Просте число)
149 (Просте число)	

9 бітні модулі:

512 (Складене число: 2 ^ 9)	373 (Просте число)
511 (Складене число: 7 * 73)	367 (Просте число)
509 (Просте число)	359 (Просте число)
505 (Складене число: 101 * 5)	353 (Просте число)
503 (Просте число)	349 (Просте число)
501 (Складене число: 167 * 3)	347 (Просте число)
499 (Просте число)	337 (Просте число)
493 (Складене число: 29 * 17)	331 (Просте число)
491 (Просте число)	317 (Просте число)
487 (Просте число)	313 (Просте число)
481 (Складене число: 37 * 13)	311 (Просте число)
479 (Просте число)	307 (Просте число)
473 (Складене число: 43 * 11)	293 (Просте число)
467 (Просте число)	283 (Просте число)
463 (Просте число)	281 (Просте число)
461 (Просте число)	277 (Просте число)
457 (Просте число)	271 (Просте число)
449 (Просте число)	269 (Просте число)
443 (Просте число)	263 (Просте число)
439 (Просте число)	257 (Просте число)
437 (Складене число: 23 * 19)	251 (Просте число)
433 (Просте число)	241 (Просте число)
431 (Просте число)	239 (Просте число)
421 (Просте число)	233 (Просте число)
419 (Просте число)	229 (Просте число)
409 (Просте число)	227 (Просте число)
401 (Просте число)	223 (Просте число)
397 (Просте число)	211 (Просте число)
389 (Просте число)	199 (Просте число)
383 (Просте число)	197 (Просте число)
379 (Просте число)	

Продовження таблиці 4.9

193 (Просте число)
191 (Просте число)
181 (Просте число)
179 (Просте число)
173 (Просте число)
163 (Просте число)
157 (Просте число)
151 (Просте число)
149 (Просте число)
139 (Просте число)
137 (Просте число)
131 (Просте число)
127 (Просте число)
113 (Просте число)
109 (Просте число)

107 (Просте число)
103 (Просте число)
97 (Просте число)
89 (Просте число)
83 (Просте число)
79 (Просте число)
71 (Просте число)
67 (Просте число)
61 (Просте число)
59 (Просте число)
53 (Просте число)
47 (Просте число)
41 (Просте число)
31 (Просте число)

10 бітні модулі:

1024 (Складене число: 2 ^ 10)
1023 (Складене число: 3 * 11 * 31)
1021 (Просте число)
1019 (Просте число)
1013 (Просте число)
1009 (Просте число)
1007 (Складене число: 53 * 19)
1003 (Складене число: 59 * 17)
997 (Просте число)
995 (Складене число: 199 * 5)
991 (Просте число)
989 (Складене число: 43 * 23)
983 (Просте число)
977 (Просте число)
973 (Складене число: 139 * 7)
971 (Просте число)
967 (Просте число)
953 (Просте число)
949 (Складене число: 73 * 13)
947 (Просте число)
941 (Просте число)
937 (Просте число)
929 (Просте число)
919 (Просте число)
911 (Просте число)
907 (Просте число)
887 (Просте число)
883 (Просте число)
881 (Просте число)
877 (Просте число)
863 (Просте число)
859 (Просте число)
857 (Просте число)
853 (Просте число)
841 (Складене число: 29 * 29)

839 (Просте число)
829 (Просте число)
827 (Просте число)
823 (Просте число)
821 (Просте число)
811 (Просте число)
809 (Просте число)
797 (Просте число)
787 (Просте число)
773 (Просте число)
769 (Просте число)
761 (Просте число)
757 (Просте число)
751 (Просте число)
743 (Просте число)
739 (Просте число)
733 (Просте число)
727 (Просте число)
719 (Просте число)
709 (Просте число)
701 (Просте число)
691 (Просте число)
683 (Просте число)
677 (Просте число)
673 (Просте число)
661 (Просте число)
659 (Просте число)
653 (Просте число)
647 (Просте число)
643 (Просте число)
641 (Просте число)
631 (Просте число)
619 (Просте число)

Продовження таблиці 4.9

617 (Просте число)	311 (Просте число)
613 (Просте число)	307 (Просте число)
607 (Просте число)	293 (Просте число)
601 (Просте число)	283 (Просте число)
599 (Просте число)	281 (Просте число)
593 (Просте число)	277 (Просте число)
587 (Просте число)	271 (Просте число)
577 (Просте число)	269 (Просте число)
571 (Просте число)	263 (Просте число)
569 (Просте число)	257 (Просте число)
563 (Просте число)	251 (Просте число)
557 (Просте число)	241 (Просте число)
547 (Просте число)	239 (Просте число)
541 (Просте число)	233 (Просте число)
523 (Просте число)	229 (Просте число)
521 (Просте число)	227 (Просте число)
509 (Просте число)	223 (Просте число)
503 (Просте число)	211 (Просте число)
499 (Просте число)	197 (Просте число)
491 (Просте число)	193 (Просте число)
487 (Просте число)	191 (Просте число)
479 (Просте число)	181 (Просте число)
467 (Просте число)	179 (Просте число)
463 (Просте число)	173 (Просте число)
461 (Просте число)	167 (Просте число)
457 (Просте число)	163 (Просте число)
449 (Просте число)	157 (Просте число)
443 (Просте число)	151 (Просте число)
439 (Просте число)	149 (Просте число)
433 (Просте число)	137 (Просте число)
431 (Просте число)	131 (Просте число)
421 (Просте число)	127 (Просте число)
419 (Просте число)	113 (Просте число)
409 (Просте число)	109 (Просте число)
401 (Просте число)	107 (Просте число)
397 (Просте число)	103 (Просте число)
389 (Просте число)	101 (Просте число)
383 (Просте число)	97 (Просте число)
379 (Просте число)	89 (Просте число)
373 (Просте число)	83 (Просте число)
367 (Просте число)	79 (Просте число)
359 (Просте число)	71 (Просте число)
353 (Просте число)	67 (Просте число)
349 (Просте число)	61 (Просте число)
347 (Просте число)	47 (Просте число)
337 (Просте число)	41 (Просте число)
331 (Просте число)	37 (Просте число)
317 (Просте число)	
313 (Просте число)	

В таблиці 4.10 розраховані на основі розробленої програми набори модулів для процесорів різної розрядності (Додаток В).

Таблиця 4.10 – Набори взаємопростих модулів та їх характеристики для процесорів різної розрядності

16 бітний процесор:

Модулі	p_i	m_i	B_i	P,
4 бітні:	16	7	105105	240240
	15	11	176176	
	13	2	36960	
	11	9	196560	
	7	6	205920	
5 бітні:				
1)	29	28	63308	65569
	19	8	27608	
	17	8	30856	
	7	1	9367	
2)	27	1	2431	65637
	17	9	34749	
	13	8	40392	
	11	9	53703	
3)	29	9	20475	65975
	25	9	23751	
	13	8	40600	
	7	5	47125	
4)	32	31	64449	66528
	27	4	9856	
	11	5	30240	
	7	3	28512	
5)	32	5	10465	66976

Продовження таблиці 4.10

	23	5	14560	
	13	10	51520	
	7	6	57408	
6 бітні:				
1)	63	46	48070	65835
	55	38	45486	
	19	11	38115	
2)	61	43	46483	65941
	47	20	28060	
	23	20	57340	
3)	53	36	44892	66091
	43	39	59943	
	29	12	27348	
4)	53	37	64343	92167
	47	18	35298	
	37	34	84694	

32 бітний процесор:

Модулі	P_I	m_i	B_I	P
5 бітні:				
1)	32	1	137894913	4412637216
	31	16	2277490176	
	29	25	3803997600	
	27	17	2778327136	
	23	14	2685953088	
	19	14	3251416896	
	13	8	2715469056	
2)	31	1	145422675	4508102925
	29	4	621807300	

Продовження таблиці 4.10

	27	8	1335734200	
	25	3	540972351	
	23	13	2548058175	
	19	15	3559028625	
	17	1	265182525	
3)	32	21	5568833025	8485840800
	31	17	4653525600	
	29	13	3803997600	
	27	11	3457194400	
	25	18	6109805376	
	23	22	8116891200	
	19	5	2233116000	
6 бітні:				
1)	64	15	1006745025	4295445440
	59	16	1164866560	
	55	27	2108673216	
	43	36	3596186880	
	37	9	1044838080	
	13	12	3965026560	
2)	59	44	3277649716	4395030301
	47	17	1589691811	
	43	32	3270720224	
	41	36	3859050996	
	31	18	2551953078	
	29	20	3031055380	
3)	59	6	436916418	4296344777
	53	45	3647839905	
	47	15	1371173865	

Продовження таблиці 4.10

	41	4	419155588	
	31	25	3464794175	
	23	19	3549154381	
7 бітні:				
1)	125	59	2027274751	4295073625
	107	11	441549625	
	83	75	3881090625	
	73	67	3942053875	
	53			
2)	119	54	1951010118	4299448223
	89	78	3768055746	
	83	13	673407553	
	73	44	2591448244	
	67	61	3914423009	
3)	101	57	2428247253	4302683729
	89	73	3529167553	
	83	21	1088630823	
	79	73	3975897623	
	73	32	1886107936	
4)	128	83	2785025409	4294978944
	123	70	2444296960	
	101	45	1913604480	
	73	68	4000802304	
	37	15	1741207680	

64 бітний процесор:

Модулі	p_i,	m	B	ПР,
6 бітні:				
1)	53	1	349152177107071296	18505065386674778688

Продовження таблиці 4.10

	47	40	15748991818446620160	
	43	25	10758758945741150400	
	41	20	9026861164231599360	
	31	12	7163251117422494976	
	29	19	12124008356786923968	
	19	3	2921852429474965056	
	17	6	6531199548238157184	
	53	1	349152177107071296	
	47	40	15748991818446620160	
	43	25	10758758945741150400	
	41	20	9026861164231599360	
7 бітні:				
1)	113	91	45624703079436213187	566548510766625504 41
	109	83	43140849902412767767	
	107	97	51360005181647358811	
	103	95	52254474294009148465	
	101	52	29168834217687649732	
	97	49	28619460853159432697	
	89	69	43923423868423775061	
	83	35	23890599851604689945	
	79	49	35140350667803354071	
	73	56	43461255620453463352	
2)	128	103	393110229925203338625	488525334275980848000
	127	31	119246341437444144000	
	125	114	445535104859694533376	
	123	110	436892575368763360000	
	121	7	28261796197784016000	
	119	30	123157647296465760000	

Продовження таблиці 4.10

	113	19	82141427887111824000	
	109	27	121010862618820944000	
	107	49	223717209154421136000	
	103	99	469553476634195184000	
8бітні:				
1)	179	44	13135832020200000892	53438952991268185447
	173	28	8649079096852654292	
	167	44	14079724141411977004	
	163	11	3606309711067178159	
	157	127	43227688088478086317	
	151	27	9555309475259874219	
	149	63	22594993546643595189	
	137	58	22623790317471202598	
	131	56	22844132576419987672	
2)	181	72	241165613757449117016	606263556806920696943
	191	114	361853641235544290322	
	193	49	153921835666005772799	
	197	107	329290358265687891233	
	199	185	563611849292865974545	
	211	123	353414300887446662199	
	217	165	460983810475308364035	
	223	74	201181628716197899434	
	227	137	365894745738097513133	

4.6 Розробка структури спецпроцесора визначення залишку багаторозрядного числа, його системні характеристики та мікроелектронна реалізація

Спецпроцесор визначення залишку багаторозрядних чисел належить до пристроїв опрацювання та перетворення даних і може бути використаний в

системах передавання інформації, а також захисту даних від помилок та несанкціонованого доступу.

У праці [105] показано існуючий пристрій обчислення залишку на основі згортки по непарному модулю $P = 2^n - 1$, який складається з *n*-розрядного регістру і логічних схем.

Основним недоліком даного пристрою є значна апаратна складність та певні функціональні обмеження, що унеможливлює його використання для знаходження операції обчислення залишку великорозрядних двійкових чисел в діапазоні $2^{10}..2^{30}$. Крім того, з використанням даного пристрою неможливе знаходження залишку по довільному цілочисельному великорозрядному модулю *P*.

Одним з аналогів пристрою пошуку залишку є апаратний комплекс на основі згортки унітарного коду числа, з використанням лічильника і логічних схем [106].

До недоліків розглянутого пристрою належить низька швидкодія, яка обмежує при роботі з числами великої розрядності.

Наближеним за своєю суттю та технічними характеристиками до розробленого спецпроцесора визначення залишків великорозрядних чисел є пристрій, який складається з лічильника, *n*-розрядного регістра зсуву і *k*-розрядного суматора зі зворотнім зв'язком, шини запису двійкового коду числа на основі згортки по непарному модулю [105].

Даний пристрій характеризується великою апаратною складністю, в склад якого входить багаторозрядний двійковий суматор з розрядністю вхідного двійкового коду *k*, великорозрядного модуля *P*. Слід зазначити, що використання його можливе тільки для знаходження залишку по непарному модулю.

З врахування недоліків існуючих пристроїв були поставлені задачі:

1. Розробити спеціалізований процесор пошуку залишків великорозрядних чисел з меншою апаратною складністю;

2. Розширити функціональні можливості пристрою отримання залишку великорозрядного двійкового числа по будь-якому цілочисельному багаторозрядному модулю *p*.

Поставлені задачі можна вирішувати з використанням запропонованого пристрою, в основу якого входять:

1) n-розрядний регістр зсуву, вхід якого підключений до шини запису кодового представлення числа;

2) шина запису в базисі Радемахера модуля P, підключена до першого входу регістра зсуву з k+1-розрядами. Крім того, другий вхід з'єднується з першим входом блоку управління (БУ), другий і третій виходи БУ з'єднані з першим входом третього і четвертого k+1-розрядного регістру зсуву, виходи яких відповідно підключені до першого і другого входів мультиплексора, третій вхід якого з'єднаний з четвертим виходом БУ, а вихід підключений до першого входу накопичувального суматора, другий вхід якого підключений до виходу другого k+1-розрядного регістру зсуву, а вихід з'єднаний з другим входом БУ і додатково введеною вихідною шиною кодового представлення залишку.

Функціональна схема розробленого спецпроцесора зображена на рис.4.20 [111], де:

1 –регістр зсуву з розрядністю n біт (№1);

2 – шина даних двійкового числа Y;

3 – шина даних представлення модуля P в базисі Радемахера;

4 – блок управління;

5 – регістр зсуву з розрядністю k+1-біт (№2);

6 – регістр зсуву з розрядністю k+1-біт (№3);

7 – регістр зсуву з розрядністю k+1-біт (№4);

8 – однорозрядний накопичувальний суматор;

9 – мультиплексор;

10 – вихідна шина кодового представлення залишку b_i.

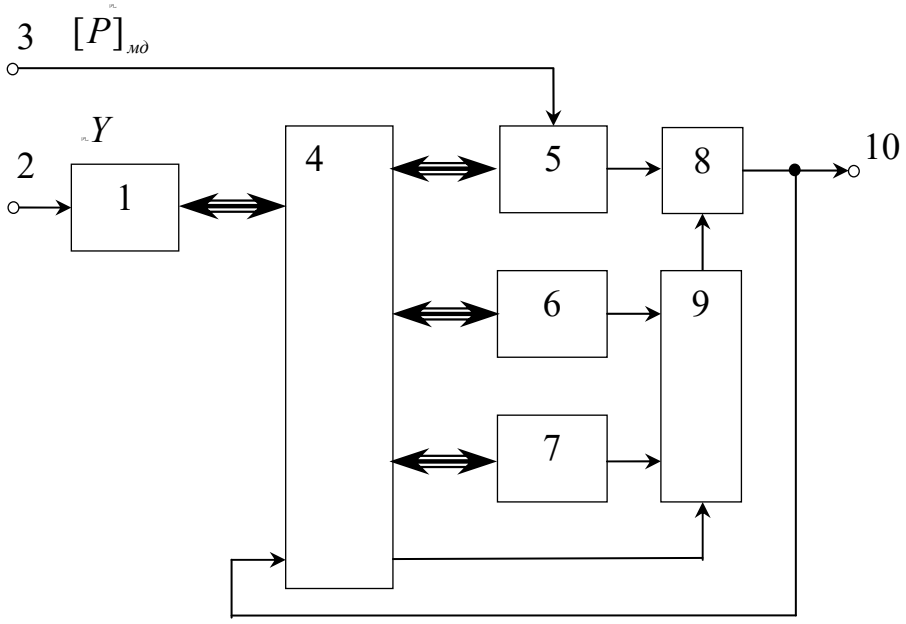


Рисунок 4.20 - Функціональна схема спецпроцесора обчислення залишку великорозрядних чисел

Робота спецпроцесора починається з циклу визначення залишку числа Y у першому n-розрядному регістрі зсуву і другому k+1-розрядному регістрі зсуву, на основі адресних сигналів БУ записується код числа Y в базисі Радемахера та $[P]_{мд}$, а в третій і четвертій k+1-розрядні регістри зсуву записуються нулі.

Після кожного наступного циклу роботи пристрою на четвертому виході БУ формується сигнал «1», що здійснює зсув на один розряд в бік старших розрядів коду залишку b_{i-1} у регістрі №3 та здійснюється запис старшого біта числа Y a_i у молодший розряд регістра №3. В результаті чого отримуємо значення $2b_{i-1} + a_i$.

В кожному циклі роботи спецпроцесора паралельно через мультиплексор на входи однорозрядного накопичувального суматора, по одному розряду зчитується код $[P]_{мд}$ розрядністю $k+1$ та код відповідного регістра №3 або №4. Крім цього, паралельно записується новий залишок b_i у відповідний регістр №4 або №3, згідно адресних входів БУ.

У результаті чого отримана за $k+1$ тактів у однорозрядному накопичувальному суматорі по кожному розряду отримується сума кодів, яка припиняється формуванням на виході суматора останнього біта «0» або «1», при чому отримане

значення подається на другий вхід БУ. Крім того, якщо біт є «1», то текучий залишок b_i у регістрі №3 задовільняє нерівність $b_i < P$ і відбувається зсув на один розряд в бік старших розрядів в регістрі №3, а в молодший розряд записується наступний молодший біт числа *Y*. Якщо на виході суматора сигнал «0», тоді $b_i \geq P$ і на вхід мультиплексора надходить сигнал «0», який поступає на четвертий вихід БУ, після чого запускається новий цикл сумування у суматорі кодів $[P]_{мд}$ регістрів №2 та №4. Після чого здійснюється запис інформації у регістр №3 до виникнення «0» біта в кінці *k*+1 такту на виході суматора. У результаті чого запускається попередній цикл.

Завершення роботи спецпроцесора здійснюється після зчитування останнього молодшого біта числа *Y* a_1 в одному з регістрів №3 або №4 і відповідно отримується b_1 в базисі Радемахера, який зчитується через мультиплексор і суматор на вихідну шину, при цьому на виході регістра №2 формується сигнал «0».

На рис.4.21 представлена структурна схема спецпроцесора обчислення залишку великорозрядних чисел.

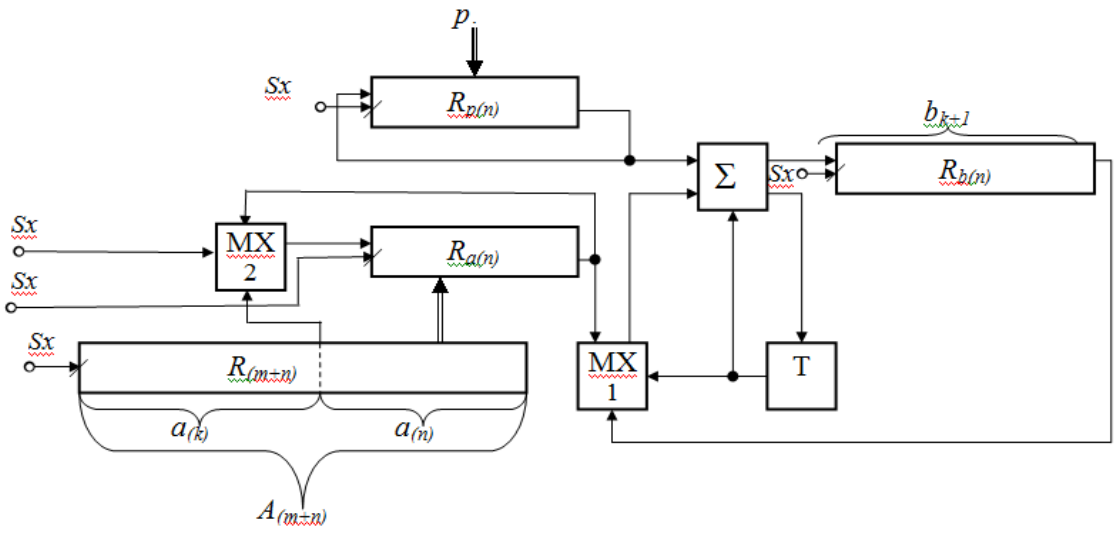


Рисунок 4.21 - Структурна схема спецпроцесора обчислення залишку великорозрядних чисел

Роботу такого спецпроцесора можна описати наступним чином:

1. В першому такті в регістр $R_{(m+n)}$.записується прямий код великорозрядного числа А, в регістр $R_{a(n)}$ записується *n*-розрядний фрагмент старших бітів числа А

початкового n-розрядного залишку a , в регістр $R_{p(n)}$ записується доповнюючий код мантиси модуля P , а регістр R_b і тригер T скидаються в нульовий стан.

2. За n тактів синхронізації виконується побітне сумування через мультиплексор MX(1) в залежності від стану тригера T кодів регістрів $R_{a(n)}$ та $R_{p(n)}$ в суматорі Σ в результаті чого в регістр R_b заноситься результат операції сумування кодів a_i+p* і одночасно через мультиплексор MX(2) код регістра $R_{a(n)}$ реверсується.

3. Після завершення n тактів циклу 2 переключається мультиплексор 2, зсувається на один розряд інформація в регістрах a і p, в результаті чого в молодший розряд регістра $R_{a(n)}$ записується старший біт $a_{(k)}$ компонента, а його попередній код подвоюється.

4. Цикли 2 і 3 повторюються до завершення розрядності числа A і в регістрі R_b формується кінцевий залишок b.

5. Виконується сумування p і a, при цьому в регістр R_{b+1} записуються нові залишки або його доповнення, а в регістр R_a подвоєння значення a.

6. У залежності від знакового тригера з p. сумується з $2a$ або b_{k+1} (значення регістру R_b), після цього цикл повторюється m раз.

Розроблений спецпроцесор складається з двох регістрів на основі флеш-пам'яті, двох цифрових компараторів, суматора, регістра зсуву та блоку управління (табл.4.11).

Таблиця 4.11 - Характеристика спецпроцесора обчислення залишків

Компоненти	Кількість, шт.	Коефіцієнт апаратної складності C_A, експлуат. один.	Кбіт
Флеш-память	2	10	1
Цифровий компаратор	2	1	0,01
Суматор	1	2	0,01
Блок управління	1	5	0,1
Регістр зсуву	1	3	0,03

Виходячи з вищенаведених характеристик пристроїв обчислення залишків по модулю, у (4.3)-(4.5) наведено аналітичні вирази швидкодії розглянутих пристроїв (рис. 4.21):

$$S = 1/(2(n-1)), \tag{4.3}$$

де $S(n)$ - кількість залишків з розрядністю n процесор обчислює в унітарному коді за 1 с.

$$S1 = p/2^n, \tag{4.4}$$

де $S1(n)$ - кількість залишків з розрядністю n процесор обчислює в двійковому коді базису Радемахера за 1 секунду, p - модуль.

$$S2 = 4^{20-n}, \tag{4.5}$$

де $S2(n)$ – кількість залишків з розрядністю n процесор шукає в базисі Крестенсона за 1 секунду, $i = 5\ldots9$.

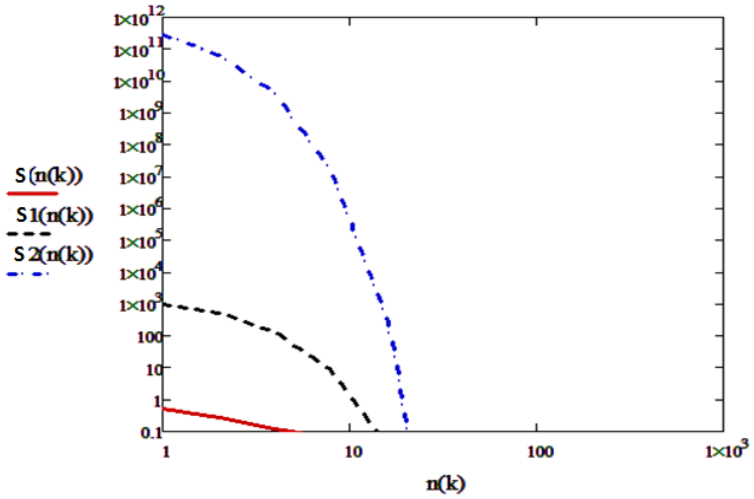


Рисунок 4.21 - Швидкодія пристроїв обчислення залишків по модулю

З результатів дослідження видно, що запропонований спецпроцесор дозволяє знаходити найбільшу кількість залишків за 1 с в порівнянні з відомими аналогами.

При дослідженні апаратної складності запропонованого спецпроцесора слід врахувати дані табл. 4.11. Тоді апаратна складність буде обчислюватися згідно наступного співвідношення:

$$P(n) = F(Ф + ЦК + \sum + РЗ + БУ), \tag{4.6}$$

де Ф - апаратна складність флеш-пам’яті;

ЦК - апаратна складність цифрового компаратора;

$\sum$ - апаратна складність суматора;

РЗ – апаратна складність регістру зсуву;

БУ – апаратна складність блоку управління.

Оскільки ЦК+∑+РЗ+БУ= C_A=const, тоді загальна апаратна складність з врахуванням коефіцієнту апаратної складності флеш-пам'яті буде:

$$P(n) = F \cdot \log_2 n + C_A, \tag{4.7}$$

де n - розрядність числа.

На рис. 4.22 подано результати дослідження при використанні елементної бази різних виробників.

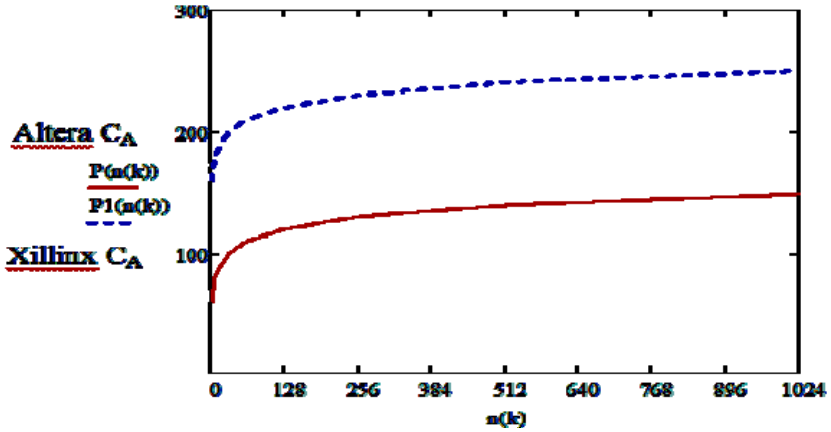


Рисунок 4.22 - Апаратна складність запропонованого методу при використанні елементної бази різних виробників.

Результати проведеного експерименту показали, що використання елементної бази фірми Xilinx при проектуванні спецпроцесора обчислення залишків по модулю дозволяє зменшити апаратну складність на 20% в порівнянні з елементною базою фірми Altera. Завдяки наявності в платформі PCI, що дозволяє імплементовувати її в існуючі обчислювальні системи, як окремий елемент. Перевагою даної платформи є сумісне використання ПЛІС XilinxSpartanIII та AlteraCyclone в яких доступ до оперативної пам'яті SRAM розміром 16Мб спільний, що дає можливість провести аналіз системних характеристик спецпроцесора на різних кристалах.

ВИСНОВКИ ДО РОЗДІЛУ 4

1. Розроблені та досліджені системні та схемо-технічні характеристики компонентів спецпроцесорів у базисі Крестенсона, що дозволило оптимізувати часову та апаратну складність базових вузлів великорозрядних спецпроцесорів.

2. Встановлено функціональні переваги апаратної та часової складності модульних матриць сумування, множення та шифрування даних у базисі Крестенсона по відношенню до базису Радемахера, що дозволяє ефективно використати досліджувані компоненти при побудові високопродуктивних спецпроцесорів опрацювання великорозрядних чисел у базисі Крестенсона.

3. Розроблені алгоритми міжбазисного перетворення Радемахера-Крестенсона на основі комутованих рандомізаторів шляхом глибокого розпаралелення процесів опрацювання бітів великорозрядних чисел базису Радемахера, що дозволило досягнути максимальної швидкодії перетворення та отримати всі коди залишків n, k-розрядного перетворення Радемахера-Крестенсона з глибоким розпаралеленням.

4. Розроблений новий алгоритм та структура спецпроцесора міжбазисного перетворення Крестенсона-Радемахера шляхом застосування досконалої нормалізованої форми СЗК та матричної структури спецпроцесора на суматорах з наскрізним переносом, що дозволило за рахунок розпаралелення операцій сумування виконати міжбазисне перетворення з часовою складністю $(kn/2)\nu$.

5. Розроблено програмне забезпечення та розраховані таблиці систем взаємно простих модулів для процесорів з різною розрядністю та різною розрядністю кодів модулів, що дозволило оптимізувати характеристики перетворень системи залишкових класів у великорозрядних спецпроцесорів базису Крестенсона та на мові VHDL.

6. Результати чисельного експерименту показали, що використання елементної бази фірми Xilinx при проектуванні спецпроцесора обчислення залишків по заданому модулю дозволяє зменшити апаратну складність на 20% порівняно з елементною базою фірми Altera. Платформа оснащена модулем інтерфейсу PCI, що

дозволяє імплементовувати її в існуючі обчислювальні системи як окремий елемент. Особливістю даної платформи є сумісне використання ПЛІС XilinxSpartanIII та AlteraCyclone, які мають доступ до спільної оперативної пам’яті SRAM розміром 16Мб, що дозволяє провести дослідження та аналіз системних характеристик спецпроцесора на різних кристалах.

ЗАГАЛЬНІ ВИСНОВКИ ПО РОБОТІ

1. Виконана систематизація процесорів цифрового опрацювання даних на основі різних архітектур та використання різних теоретико-числових базисів, що дозволило обґрунтувати перспективу розробки та реалізації високопродуктивних спецпроцесорів опрацювання великорозрядних чисел у системі числення залишкових класів базису Крестенсона.

2. Виконаний аналіз характеристик арифметико-логічних операцій у базисах Радемахера та Крестенсона, здійснено порівняння функціональних можливостей процесорів та побудовані діаграми часових затрат виконання базових операцій, що дозволило встановити основні переваги процесорів Крестенсона при вирішенні задач шифрування інформаційних потоків у сучасних комп'ютерних системах та сформулювати завдання і постановку задачі дисертаційного дослідження.

3. Викладені теоретичні засади прямого та зворотнього перетворення цілочисельної системи залишкових класів базису Крестенсона. Поданий приклад розрахунку базисних чисел B_i та виконання арифметико-логічних операцій у базисах Радемахера та Крестенсона, в результаті чого встановлено, що операції сумування та множення в базисі Крестенсона можуть виконуватись за один такт роботи процесора, а при зростанні розрядності опрацьовуваних чисел більше 128 біт швидкодія процесорів базису Крестенсона перевищує відповідні характеристики процесорів базису Радемахера на 2-3 порядки.

4. Розроблено алгоритм операції ділення на основі бінарно-розмежованої СЗК та отримано аналітичну матрицю цієї операції у базисі Крестенсона, що дозволило замінити операцію ділення операцією множення в розширеній СЗК та доповнюючими кодами залишків і виконувати її за обмежене число тактів процесора з підвищенням швидкодії виконання операції ділення на 2-3 порядки.

5. Розроблені принципи та теоретичні основи розмежування розрядної сітки у базисі Радемахера-Крестенсона, у результаті чого отримані аналітичні вирази прямого та зворотнього перетворень розмежованої системи. Показано, що при

бінарному розмежуванні базису Радемахера формуються матриці степеневих залишків двійкових чисел, які дозволяють зменшити на 2-3 порядки алгоритмічну та часову складність міжбазисних перетворень, а також операцій додавання, множення та піднесення до степеня великорозрядних двійкових чисел.

6. Отримані аналітичні вирази та досліджена часова складність алгоритмів міжбазисних перетворень Радемахера-Крестенсона та Крестенсона-Радемахера. Розрахована та побудована діаграма алгоритмічної складності досліджених міжбазисних преретворень.

7. Розроблені теоретичні засади та алгоритми виконання міжбазисних перетворень на основі матричних суматорів у розмежованій СЗК пірамідального та лінійного типу, в результаті встановлено, що в бінарно-розмежованій СЗК підвищується ефективність реалізації міжбазисного перетворення Радемахера – Крестенсона за схемотехнічними варіантами на основі пірамідально та лінійно з'єднаних суматорів по модулю P, при чому, об'єм обладнання пірамідальної структури в два рази перевищує об'єм лінійної структури при заданій розрядності процесора. Швидкодія пірамідального МБП із збільшенням розрядності процесора від 8 до 128 зростає відповідно від 1,6 до 13,8 разів.

8. Запропоновані рекурентні алгоритми та високопродуктивні спецпроцесори опрацювання великорозрядних чисел у базисі Радемахера-Крестенсона шляхом використання однорозрядного двійкового суматора та ПЗП в якості обчислювача поточних сум залишків по модулю, що дозволило зменшити на порядок апаратну складність та підвищити на 2 порядки швидкодію спецпроцесорів даного класу.

9. Розроблені та досліджені системні та схемотехнічні характеристики компонентів спецпроцесорів у базисі Крестенсона. Встановлено функціональні переваги апаратної та часової складності модульних матриць сумування, множення та шифрування даних у базисі Крестенсона по відношенню до базису Радемахера.

10. Розроблено високопродуктивний спецпроцесор міжбазисного перетворення Радемахера-Крестенсона, що, у порівнянні з відомими аналогами, шляхом мультиплексування та рандомізації розмежованих залишків дозволило досягнути максимальної швидкодії визначення кодів залишків, у системі

взаємопростих модулів, з часовою складністю перемикання двох послідовно з’єднаних логічних вентилів, незалежно від розрядності перетворюваного двійкового числа.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Трахтман А.М. Основы теорий дискретных сигналов на конечных интервалах /А.М. Трахман, В.А. Трахман – М.: Сов. радио, - 1978.

2. Залмазон Л.А. Преобразование Фурье, Уолша, Хаара и их применение в управлении, связи и других областях/Л.А. Залмазон –М.: Наука, 1989. – 496 с.

3. Бекчанова Ш.Б. Алгоритмы и структуры на основе быстрых преобразований Хаара / Ш.Б. Бекчанова, Х.Н. Зайниддинов // Техника юлдузлари. – Ташкент, –2002. –№4, С.45-54.

4. Задірака В.К. Комп'ютерна арифметика багаторозрядних чисел: Наукове видання / В.К. Задірака, О.С. Олексюк //– Київ. –2003. – 264 с.

5. Николайчук Я.М. Теорія цифрових перетворень мультибазисного супершвидкодіючого процесора /Я.М. Николайчук// Искусственный интеллект. – 2008. – №4. – С. 387-394.

6. Палагин А.В. Реконфигурируемые структуры на ПЛИС / А.В. Палагин, В.Н. Опанасенко, В.Г. Сахарин // УсиМ. – 2000. – № 3. – С. 33-43.

7. Мельник А О. Програмовані процесори обробки сигналів. – Львів: Вид-тво Національного університету "Львівська політехніка", –2000. –55 с

8. Задірака В.К. Комп'ютерна криптологія: Підручник /В.К. Задірака, О.С. Олексюк // – Київ: – 504 с.

9. Майоров С.А. Принципы организации цифровых машин / С.А. Майоров, Г.И. Новиков// – Л.: Машиностроение, –1974. – 306 с.

10. Корнейчук В.И. Основы компьютерной арифметики / В.И. Корнейчук, В.П. Тарасенко // – К.: Вища школа. –2003. - С. 34-56.

11. Мельник А.О. Спеціалізовані комп'ютерні системи реального часу /А.О. Мельник// – Львів: Державний університет "Львівська політехніка", 1996. – 54 с.

12. Самофалов К.Г. Цифровые ЭВМ / К.Г. Самофалов, В.И. Корнейчук, В.П. Тарасенко// – СПб.: Вища школа – К. – 2000. – 528с.

13. Стешенко В.Б. ПЛИС фирмы «Altera»/ В.Б.Стешенко//. – М.: «Додетка» –2002. – 575с.

14. Зотов В.Ю. Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы Xilinx / В.Ю. Зотов/ – М.:Телеком. –2006. – 522с.

15. Палагин А.В. Опыт разработки микропроцессорных распределенных систем риального времени./ А.В. Палагин, Я.М. Николайчук// – Киев: Знание, – 1988. – 19 с.

16. Офіційний сайт.Intel microcontrollers [Електронний ресурс] – Режим доступу: URL: http://www. Intel.com/design/embrocontrol/index.htm.

17. Архітектура системи Motorola Canopy [Електронний ресурс] – Режим доступу URL: http://rtm.ru/canopy.html. - Назва з титул. екрану.

18. Зотов В.Ю. Проектирование встраиваемых микропроцессорных систем на основе ПЛИС фирмы XILINX / В.Ю. Зотов / М:., Горячая Линия. –Телеком, – 2006. –522с.

19. Стешенко В.Б ПЛИС фирмы «Altera» / В.Б. Стешенко // М., «Додека». – 2002. –575с.

20. Тарасов И.Е. Разработка цифровых устройств на основе ПЛИС XILINX с применением языка VHDL. /И.Е. Тарасов// М.: Горячая линия. –Телеком, –2005. – 782с.

21. Stallings W. Computer Organization and Architecture / W. Stallings// 5th ed., New York, NY: Macmillan Publishing Company Stallings –2000.

22. Tanenbaum A., Structured Computer Organization / A. Tanenbaum// 4th ed. Upper Saddle River, NJ: Prentice Hall –1999.

23. Patterson D. Computer Architecture. A Quantitative Approach / D. Patterson, J. Hennessy // Morgan Kaufmann Publishers, Inc –1996.

24. Якименко І.З. Розмежована система числення залишкових класів та спецпроцесори на її основі // І.З. Якименко, О.І. Волинський / Поступ в науку. Збірник наукових праць Бучацького інституту менеджменту і аудиту. - 2010. - №6, Т1. – С.80-83.

25. Акушский И.Я. Машинная арифметика в остаточных класах / И.Я. Акушский, Д.И. Юдицкий – М: Сов.радио, 1968. – 440 с.

26. Брюховин Е.Н. О принципе построения компьютеров, специализированных на вычислении некоторого набора специальных функций в мультипроцессорной вычислительной системе / Е.Н. Брюховин // Автоматика и вычислительная техника, 1983, № 12. — С. 45-51.

27. Торгашев В.А. Система остаточных классов и надежность ЦВМ. – М.:Советское радио, 1970 – 118 с.

28. Николайчук Я.М. Теорія джерел інформації / Я.М. Николайчук – Тернопіль: ТзОВ „Терно–граф”, 2010. – 536 с.

29. Червяков Н.И. Нейрокомпьютеры в остаточных класах / Н.И. Червяков, П.А. Сахнюк, А.В. Шапошников, А.Н. Макоха - М.: Радиотехника, 2003. - 272 с.

30. Синьков М.В. Непозиционные представления в многомерных числовых системах / М.В. Синьков, Н.М. Губарени - Киев: Наукова думка, 1979. - 137 с.

31. Мельник А.О. Архітектура комп’ютера: Наукове видання / А.О. Мельник – Луцьк: Волинська обласна друкарня – 2008. – 470с.

32. Петришин Л.Б. Теоретичні основи перетворення форми та цифрової обробки інформації в базисі Галуа / Л.Б. Петришин// Навч. посібник. - Київ.: ІЗіМН МОУ, - 1997. - 237 с.

33. Николайчук Я.М. Проектування спеціалізованих ком’ютерних систем / Я.М. Николайчук, Н.Я. Возна, І.Р. Пітух - навч. посібник для втузів. - Т.: Тено-граф, 2010. - 392 с.

34. Николайчук Я.М. Коди поля Галуа: теорія та застосування / Я.М. Николайчук // Монографія - Тернопіль: ТзОВ «Терног раф» - 2012. – 576с.

35. Лабунец В.Г. Теоретико-числовые преобразования над полями алгебраических чисел. – В кн.: Применение ортогональных методов при обработке сигналов и анализе систем. – Свердловск: УПИ, 1981, С.44-54.

36. Бекчанов Ш.Б. Синтез быстродействующих спецпроцессоров Хаара на основе матричной диаграммы / Ш.Б. Бекчанов, Х.Н. Зайнидинов – тезис докл. НТК «Молодьож в развитии науки и техники» - Ташкент, 2002 – 78с.

37. Волинський О. Систематизація характеристик теоретико-числових базисів та їх застосування для побудови високопродуктивних спецпроцесорів/

О. Волинський, В. Пуюл // Вісник Тернопільського національного технічного університету. – 2011. – Том 16, №3. – С.183-189

38. Грибанов Ю.И. Автоматические цифровые корреляторы./ Ю.И. Грибанов, Г.П. Веселова, В.Н Андреев // – М.: Энергия, 1971. – 240с.

39. Албанський І.Б. Спецпроцесори кореляційної обробки сигналів /Албанський І.Б., Заведюк Т.О. // Праці міжнародного симпозіуму ПИТАННЯ ОПТИМІЗАЦІЇ ОБЧИСЛЕНЬ (ПОО-XXXV) Том 1. К.: - 2009.- С.8-13.

40. Ширмовська Н.Г. Оптимальна дискретизація заданих кореляційною функцією сигналів / Методи та прилади контролю якості.- Івано-Франківськ.- 2009.- №22.- С.107-111.

41. А.С. №1282160. Многоканальное устройство для вычисления структурной функции / Я.М. Николайчук – Бюлетень №1.-1987

42. А.С. №1317455. Многоканальное устроиство для вычесления функцій эквивалентности / Я.М. Николайчук С.М. Ищеряков – Бюллетень №22.-1987

43. Кожем'яко В.П. Ефективність функціонування оптико-електронних інтелектуальних структур в системах автоматизованого управління / В.П. Кожем'яко, А.А. Яровий // Вестник Херсонського государственного технического университета. – 2001. - №1(10). - С. 279-281.

44. Календер В. Компьютерная томография / В. Календер // Основы, техника, качество изображений и области клинического использования: фонография. – М .: Техносфера. - 2006. - 344с.

45. Орнатский П.П. Теоретические основы информационно-измерительной техники / 2-е изд., перераб., и доп / П.П. Орнатский – К.: Вища школа. – 1983. – 455с.

46. Explore Zilog's Products. - [Електронний ресурс]. - Режим доступу: http://www.zilog.com/products.

47. Padovani R. Reverse Link Performance of IS-95 Based Cellular Systems. IEEE Personal Communications / Third Quarter // Padovani R., 1994. - P. 28 - 34.

48. Advanced Micro Devices, AMD - Processor Homepage [Електронний ресурс]. - Режим доступу: http://amd.com.

49. RF Solutions and Wireless Communications Technology: RFMD [Електронний ресурс]. - Режим доступу: http://www.rfmd.com.

50. Шевчук Б. М. Технологія багатофункціональної обробки і передачі інформації в моніторингових мережах. / Б.М. Шевчук, В.К. Задірака, Л.О. Гнатів, С.В. Фраєр // НАН України, Ін-т кібернетики ім. В. М. Глушкова. – К. : Наук. Думка, 2010. – 371 с.

51. Задирака В.К. Методи захисту фінансової інформації: Навчальний посібник / В.К. Задирака. Олексюк О С. - Тернопіль: "Збруч", 2000. - 460с.

52. Пат. АС №1462477 СССР, МКИ H03 M1/38. “Аналого-цифровой преобразователь”/ Я.Н. Николайчук – Опубл. 28.02.89, Бюл. №8.

53. Пат. АС 13726221 СССР, МКИ H03 M1/38. “Аналого-цифровой преобразователь”/Я.Н. Николайчук. – Опубл. 07.02.88, Бюл. №5.

54. Николайчук Я.М. Дослідження системних характеристик двомірних кодів з особливими кореляційними властивостями/ Я.М. Николайчук, О.М. Заставний //Вісник технологічного університету Поділля –Хмельницький, 2004. – №2. – ч.1, Т2.

55. Николайчук Я.М. Методы цифровой обработки шумоподобных сигналов на основе кодовых ключем / Я.М. Николайчук, Б.М. Шевчук – Киев, Сб. тр. ИКАН УССР, 1988.

56. Заставний О.М Теорія та принципи побудови спецпропесора на основі базисів Радемахера, Крестенсона, Галуа. / О.М. Заставний, Я.М. Николайчук, Н.Д. Круцкевич, Р.І. Король // Тези доповідей сьомої міжнародної науково - технічної конференції – Вінниця: «УНШЕРСУМ - Вінниця», 2003 - 114с.

57. Пилипенко І.А. Дослідження методів стиснення інформації. / І.А. Пилипенко // Науковий вісник інституту менеджменту та економіки «Галицька академія» – 2006. – №2(10). – С. 78-82.

58. Яцків Н.Г. Методи стиснення в багатоканальних систем на основі кодів Галуа / Н.Г. Яцків, Я.М. Николайчук // Вісник національного університету

«Львівська політехніка». Радіоелектроніка та телекомунікації. – 2002. - №443. – С. 135-138.

59. Куприянов М., Мартынов О., Панфилов Д. Коммуникационные контроллеры фирмы Motorola. - СПб.: БХВ.-Петербург, 2001.- 560 с.

60. Бекчанова Ш.Б., Зайнидинов Х.Н. Принципы построения высокопроизводительных вычислительных структур// Тезисы докладов НТК Мафкуравий жараёнлар ва Узбекистонда фанлар ривожининг долзарб муаммолари Андижон. - 2002. – С. 441.

61. Садыхов Р.Х., Чеголин П.М., Шмерко В.П. Методы и средства обработки сигналов в дискретных базисах. – Мн.: Наука и техника, 1987. - 296 с.

62. ADSP-21000 Family Application Handbook, Vol. 1, Analog Devices, Free Download at: http://www.analog.com.

63. DSP Designer's Reference (DSP Solutions) CDROM // Analog Devices, 1999.

64. General DSP Training and Workshops: http://www.analog.com/ industry/dsp/tech_docs.html.

65. Яцків Н.Г., Спецпроцесори обробки даних на основі перетворення Крестенсона – Галуа. / Н.Г. Яцків, Р.І. Король, В.В. Яцків, Т.Г. Федчишин // Вісник Технологічного університету Поділля. – 2003. -ТІ, №3. – С. 105-108

66. Круцкевич Н.Д. Принципи побудови RCG процесора. / Н.Д. Круцкевич, Я.М. Николайчук // Тези міжнар. науково - технічної конф. “Контроль і управління в складних системах” (КУСС - 2003) – Вінниця: «УНІВЕРСУМ – Вінниця». – 2003. – С. 73.

67. Волинський О.І. Оптимізація обчислень на основі алгоритмів міжбазисних перетворень Радемахера, Крестенсона та Галуа / О.І. Волинський, О.Д. Круцкевич, П.В. Гуменний // Праці міжнародної молодіжної математичної школи “Питання оптимізації обчислень (ПОО-XXXVII)” Київ: Інститут кібернетики імені В.М. Глушкова НАН України, 2011. С. 32-33.

68. Волинський О.І. Розмежована система числення залишкових класів та спецпроцеси на її основі. / О.І. Волинський, І.З. Якименко // Поступ в науку. Збірник

праць Бучацького інституту менеджменту і аудиту – Бучач. – 2010. - №6. Т1. – С. 80-83.

69. L. L. Yang and L. Hanzo, A residue number system based parallel communication scheme using orthogonal signaling: Part II—Multipath fading channels // IEEE Trans. Veh. Technol. – 2002. - Vol. 51. – P. 1541-1553.

70. L.-L. Yang and L. Hanzo, Minimum-distance decoding of redundant residue number system codes // Proc. IEEE ICC '2001. – Helsinki (Finland) – 2001. - P. 2975-2979.

71. L.-L. Yang and L. Hanzo, Performance analysis of coded M-array orthogonal signaling using errors-and-erasures decoding over frequency-selective fading channels // IEEE J. Select. Areas Community. – 2001.- Vol. 19. - P. 211-221.

72. Lakhani G. Some Fast Residual Arithmetic Adders, // International Journal of Electronics, 1994. pp. 225-240.

73. Lenstra H. W., Divisors in residue classes // Math. Comput.- 1984. -Vol. 42. - N 165. - P.331-340.

74. Montgomery P., Modular multiplication without trial division, Math. of Comput. – 1985. Vol. 44. - N 170. - P. 519-521.

75. Шауман А. М. Основы машинной арифметики. - Л.: Изд-во Ленинградского ун-та, 1979. -457с.

76. Широчин В.П., Мухин В.Е., Ху Чженбин. Машина состояний в задачах аутентификации пользователей в компьютерных сетях // Управляющие системы и машины. - 2003. -№ 5. - С. 59-65.

77. Заставний О.М. Аналіз системних характеристик спецпроцесорів формування вихідних даних аналого-цифрових кодерів // Вісник Технологічного університету Поділля, Хмельницький, 2005, №4, ч.1, Т2. С. 223-226.

78. Николайчук Я.М. Теоретичні основи побудови та структура спецпроцесорів в базисі Крестенсона / Я.М. Николайчук, О.І Волинський, С.В.Кулина // Вісник Хмельницького національного університету. – 2007. - №3. – Т1. – С. 85-90.

79. Николайчук Я.М., Кусик Я.Б. Коди поля Галуа та їх застосування в перетворювачах форм інформації// Тезисы докладов 7-го симпозиума Проблемы создания преобразователей формы информации - Киев: ИКАН Украины. - 1992.

80. Петришин Л.Б. Николайчук Я.Н., Ищеряков С.М., Цифровая обработка сигналов на основе преобразования кодов поля Галуа // Методы и микроэлектронные средства цифровой обработки и преобразования сигналов.- Рига: ИЭВТ АН Латвии. - 1989.- С.130 - 132.

81. Петришин Л.Б. Теоретичні основи перетворення форми та цифрової обробки інформації в базисі Галуа: Навч. посібник. - Київ.: ІЗіМН МОУ, 1997. - 237 с.

82. Столлингс В. Передача данных / В. Столлингс – 4-е изд. – СПб.: Питер, 2004. – 750 с.

83. Бернард С. Цифровая связь. Теоретические основы и практичиское приминение / С. Бернард – 2-е изд.: Пер. с англ. – М.: Издательский дом «Вильянс», 2003. – 1104с.

84. Таненбаум Э. Компьютерные сети / Э. Таненбаум – 4-е изд. – Питер, 2003. – 992 с.

85. Столлингс В. Современные компьютерные сети / В. Столлингс – СПб.: Питер, 2003. – 783 с.

86. Касянчук М.М. Теорія алгоритмів RSA та Ель-Гамаля в розмежованій системі числення Радемахера-Крестенсона / М.М. Касянчук, І.З.Якименко, О.І. Волинський, І.Р. Пітух // Вісник Хмельницького національного університету, Видавництво Хмельницького національного університету. – 2011. – № 3(177). – С.265-272.

87. Николайчук Я.М. Швидкодіючий алгоритм та процесор порівняння чисел у системі залишкових класів / Я.М. Николайчук, О.І Волинський, С.В.Кулина // Науково-теоретичний журнал "Искусственный интеллект". ІПШІ МОН і НАН України "Наука і освіта". – 2008. – №3. – С.348-352.

88. Волинський О.І. Методи порівняння та сумування в розмежованій системі числення // Поступ в науку. Збірник наукових праць Бучацького інституту менеджменту і аудиту. – Бучач. – 2009. - №5. Т1. – С.91-94.

89. Албанський І.Б. Дослідження системних характеристик цифрових пристроїв множення реалізованих в різних теоретико-числових базисах / І.Б. Албанський, О.І. Волинський // Вісник Хмельницького національного університету. – 2012. – №2. – С. 179-186.

90. Николайчук Я.М., Якименко І.З.,Воронич А.Р., Волинський О.І. Пристрій визначення залишку багаторозрядного числа // Патент на корисну модель № 68872. МПК G 06 F7/00. Опубл. 10.04.2012. Бюл. № 7

91. Николайчук Я.М., Волинський О.І. Спосіб визначення залишку двійкового числа // Патент на корисну модель № 74576. МПК G 06 F5/00. Опубл. 12.11.2012. Бюл. № 21

92. Романец Ю.В. Защита информации в компьютерных системах и сетях / Под ред. В.Ф.Шаньгина. / Романец Ю.В., Тимофеев П.А., Шаньгин В.П. – М.: Радио и связь, 1999. – 328с.

93. Kinakh Y.I Reliability of Schoof algorithm and its computational complexity. Proceedings of the Xth International Conference “The Experience of Designing and Application of CAD Systems in Microelectronics”. // Kinakh Y.I, Iakymenko I.Z./ – Lviv-Polyana. – 2009. – C. 107.

94. M.Kasyanchuk Matrix Algorithms of Processing of the Information Flow in Computer Systems Based on Theoretical and Numerical Krestenson’s Basis. Proceedings of the X–th International Conference ”Modern Problems of Radio Engineering, Telecommunications and Computer Science” (TCSET–2010). // M.Kasyanchuk, I.Yakymenko, Ya. Nykolaychuk/–L’viv–Slavske.– 2010. – P.241.

95. Николайчук Я.М., Якименко І.З., Воронич А.Р.,Волинський О.І. Пристрій визначення залишку багаторозрядного числа. Патент на корисну модель № 68872. МПК G 06 F7/00. Опубл. 10.04.2012. Бюл. № 7.

96. Волинський О.І. Швидкодія міжбазисних перетворювачів Радемахера-Крестенсона./ О.І.Волинський // Збірник матеріалів проблемно-наукової міжгалузевої конференції "Юриспруденція та проблеми інформаційного суспільства"(ЮПІС - 2011))" Івано-Франківськ, 2011. С.71-75.

97. Николайчук Я.М., Волинський О.І. Спосіб визначення залишку двійкового числа. Патент на корисну модель № 74576. МПК G 06 F5/00. Опубл. 12.11.2012. Бюл. № 21

98. Волинський О.І. Теорія, алгоритми та спецпроцесори міжбазисних перетворень Радемахера-Крестенсона./ О.І.Волинський // Поступ в науку. Збірник наукових праць Бучацького інституту менеджменту і аудиту . – Бучач. - 2012. - №8. С. 50-54.

99. Montgomery P. L."Modular multiplication without trial division"/ P. L. Montgomery/ Math. Computation, vol. 44, 1985.-pp.519 – 521

100. Волинський О. І., «Методи міжбазисних перетворень на основі розмежованої системи числення залишкових класів», Вісник національного університету "Львівська політехніка" , "Комп'ютерні системи та мережі"– Львів. – 2010. - № 688. - С.53-59.

101. Червяков Н.И. Преобразователь десятичного кода в код системы остаточных классов// А.С. СССР №374595 Бюллетень №15. – 1973

102. Червяков Н.И. Устройство для преобразования чисел из десятичной системы счисления в систему остаточных классов// А.С. СССР №377767 Бюллетень №18. – 1973

103. Болтков А.П., Червяков Н.И., Хлевнов С.Н. Устройство для преобразования чисел из позиционной системы счисления в систему остаточных классов// А.С. СССР №1008729. Бюллетень №12. – 1983

104. Хетагуров Я.А., Руднев Ю.П. Повышение надежности цифровых устройств методами избыточного кодирования. - М.: Энергия, 1974. – 272с.

105. Новиков Л.Г., Шурыгин И.Т. Счётчики импульсов с коэффициентами счёта, управляемыми с помощью двоичного кода. Журнал «Приборы и системы управления» № 6, 1972. –С.30-31

106. Авт. свідоцтво СРСР №206908, кл. G 06 F 15/34.

107. Авт. свідоцтво СРСР №337784, кл. G 06 F 15/34. Бюлетень № 15. Опубліковано 05.05.1972.

108. Авт. свідоцтво СРСР №840924, кл. G 06 F 15/36. Бюлетень №23. Опубліковано 23.06.1981.

109. Малашевич Б. М. Неизвестные модулярные суперЭВМ// PC WEEK/RE, М., 2005. № 9. С. 44-45. № 10. – С. 52-54.

110. Волинський О.І. Методи високопродуктивних перетворень великорозрядних чисел з базису Радемахера у базис Крестенсона / О.І. Волинський // Вісник національного університету “Львівська політехніка”, “Комп’ютерні системи та мережі”– Львів. – 2012. - С.39-48.

111. Николайчук Я.М., Волинський О.І. Пристрій для перетворення чисел з позиційної системи в систему залишкових класів. Патент на корисну модель № 76623 МПК G06F5/02 Опублікований 10.01.2013 Бюл.№1.

112. Николайчук Я.М., Албанський І.Б., Волинський О.І. Цифровий авто корелятор. Патент на корисну модель № 76622. МПК G 06F 17/15 Опубл. 10.01.2013 Бюл.№1.

113. Николайчук Я.М. Теорія побудови та компоненти швидкодіючих процесорів на основі досконалої та розмежованої форм системи залишкових класів / Я.М. Николайчук, О.І Волинський, С.В.Кулина // Поступ в науку. Збірник праць Бучацького інституту менеджменту і аудиту.– Бучач.– 2008.– №4. - Т1.– С.31-36.

114. Николайчук Я.М. Теорія та техніка високопродуктивних мультибазисних процесорів / Я.М. Николайчук, О.Д.Круцкевич, С.В.Кулина, О.І.Волинський // Праці міжнародного симпозіуму. Питання оптимізації обчислень. (ПОО - XXXV) . - Київ. – 2009.– Т2. – С.165-169

115. Orest Volinskiy. Methods of interbase transformations are on the basis of the delimited scale of notation of remaining classes / Orest Volinskiy // Proceedings of the 4-th International conference “Advanced Computer Systems and Networks: Design and Application” (ACSN-2009). – Lviv. - 2009. – №4. –C.314-317.

116. Orest Volinskiy. An algorithm of calculation of degrees of numbers is in the delimited system of remaining classes(DSRC) /O.I. Волинський// Proceedings of the X-th International conference “Modern Problems of Radio Engineering, Telecommunications and Computer Science” (TCSET-2010). – 2010. – №10. –p.305.

117. Orest Volynskyy. Multibases special processor module and correlations processing of information flows / Orest Volynskyy, Ivan Albanskiy, Petro Humenniy, Ostap Krutskevych, Volodymyr Puyul // Proceedings of the XIth International conference“The experience of designing and application of CAD systems in micro-electronics.– 2011. - №11 – pp.176-177.

118. Волинський О.І. Методи ділення великорозрядних чисел в теоретико-числовому базисі Радемахера-Крестенсона / О.І.Волинський // Поступ в науку. Збірник наукових праць Бучацького інституту менеджменту і аудиту . – Бучач. – 2011. – №7. Т1. –С.37-39.

119. Івасьєв С.В. Метод знаходження залишків велико-розрядних чисел в базисі Радемахера / С.В. Івасьєв, О.І. Волинський // Поступ в науку. Збірник наукових праць Бучацького інституту менеджменту і аудиту . – Бучач. – 2011. – №7. Т1. –С.88-91.

120. Yaroslav Nykolaychuk. Rademacher-Krestenson’s method of between-bases transformations in designing processors / Yaroslav Nykolaychuk, Orest Volynskyy, Andrii Borovyi // Proceedings of the 6th International Conference “Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications”. –Prague, Czech Republic. – 2011. – pp.310-313.

121. Ivan Albanskiy. Structure and Simulation of Interactive Computer Systems Based on Multibasises Switching Processors / Ivan Albanskiy, Petro Humenniy, Orest Volinskiy, Tanya Zavedyuk // Proceedings of the XI-th International conference “Modern

Problems of Radio Engineering, Telecommunications and Computer Science" (TCSET-2012). – Lviv-Slavsk. – 2012. – p.434.

122. Tsanko R.. Theory, Topology and Building Technology of Multibasis Specialized Processor / R. Tsanko, O. Volynskyy, V. Puyul, I. Pituh // Proceedings of the XI-th International conference "Modern Problems of Radio Engineering, Telecommunications and Computer Science" (TCSET-2012). – Lviv-Slavsk. – 2012. – p. 260.

123. Omondi and B. Premkumar,2007, "Residue Number System- theory and implementation," Imperial College Press.

124. M. Hosseinzadeh, S. J. Jassbi and k. Navi, 2007, "A Novel Multiple Valued Logic OHRNS Modulo rn Adder Circuit," International Journal of Electronics, Circuits and Systems, Vol. 1, No. 4, pp. 245-249.

125. M. Abdallah and A. Skavantzos, On MultiModuli residue number systems with moduli of forms ra, rb-1, rc+1 "IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS-I: REGULAR PAPERS, VOL. 52, NO. 7 , 2005.

126. F. J. Taylor, "Residue arithmetic: A tutorial with examples," IEEE Comput. Mag., vol. 17, pp. 50–62, May 1984.

127. F.J. Taylor,"Large Moduli Multipliers for Signal Processing,"IEEE Trans. Circuits Syst., vol. CAS-28, pp. 731-736, July 1981.

128. G.A. Jullien, "Implementation of Multiplication, Modulo a Prime Number, with Applications to Number Theoretic Transforms," IEEE Trans. Comput., vol. C-29, pp. 899-905, Oct. 1980.

129. W. A. Chren, Jr., C. H. Brogdon and D. Andrevska, "Delay Power Product Simulation Results For One-Hot Residue Number System Arithmetic Circuits.", IEEE, 1997. Pp. 544-547

130. H. Garner, "The Residue Number System," IEEE Transactions Electronic Computer, Vol. 8, pp.140-147, 1959.

131. N. Szabo and R. Tanaka, Residue arithmetic and its applications to computer technology, (New York, McGraw-Hill, 1967).

132. M. Hosseinzadeh, K. Navi and S. Timarchi, "Design Residue Number System Circuits in Current mode," 14th Iranian Conference of Electrical Engineering, 2006.

133. M. Hosseinzadeh, K. Navi and S. Timarchi, “New Design of 4-3 Compressor,” 11th International CSI Computer Conference of Iran, 2006.

134. K.A. Gbolagade,A Memoryless MRC Technique for RNS-to-Binary Conversion Using The Moduli Set (2^n,2^n-1, 2^{n-1}-1), International Journal of Soft Computing, Vol. 4(3),2009, pp. 127-130.

135. H. Siewobr, K. A. Gbolagade (2011): “An Efficient RNS Overflow Detection Algorithm” (in press), Far EastJournal of Electronics and Communications.

136. K.A.Gbolagade,S. D. Cotofana,Residue Number System Operands to Decimal Conversion for 3-Moduli Sets, Proceedings of 51st IEEE Midwest Symposium on Circuits and Systems (MWSCAS 08), Knoxville, USA,2008, pp. 791-794.

137. K.A.Gbolagade,S. D. Cotofana, MRC Technique for RNS to Decimal Conversion Using the Moduli Set {2^n +2, 2^n + 1, 2^n}, Proceedings of the 16th Annual Workshop on Circuits, Systems and Signal Processing.,Veldhoven, The Netherlands, 2008, pp. 318-321.

138. K. A.Gbolagade, S. D. Cotofana, Residue-to-Decimal Converters for Moduli Sets with Common Factors, Proceedings of 52nd IEEE International Midwest Symposium on Circuits and Systems, (MWSCAS 2009),Cancun, Mexico, 2009, pp. 624-627.

139. Leonel Sousa, Efficient Method for Magnitude Comparison in RNS Based on Two Pairs of Conjugate Moduli. Electrical and Computer Engineering Department, INESC-ID/IST, TU Lisbon.

140. Theodore L. Residue Addition Overflow Detection Processor Boeing Company, Seattle, Wash.Appl. No.: 414276, Sep. 29, 1989.

141. Theodore L. Houk, Seattle, Wash, Method And Apparatus For Pipelined Detection Of Overflow In ResidueArithmetic Multiplication, Boeing Company, Seattle, Wash. Appl. No.: 472,237, Jan. 30, 1990.

142. Mehrin Rouhifar, Mehdi Hosseinzadeh and Mohammad Teshnehlab, “A new approach to Overflowdetection in moduli set (2^n-1, 2^n, 2^n+1), International Journal of Computational Intelligence and InformationSecurity, Vol.2, No. 3,March, 2011, pp. 35-43.

143. Tadeusz Tomczak. Hierarchical residue number systems with small moduli and simple converters. International Journal of Applied Mathematics and Computer Science. Volume 21, Issue 1, ,2011, pp. 173–192

144. H. Siewobr, K.A.Gbolagade. Overflow detection in residue number systems addition beforeforward conversion. International Journal of Computational Intelligence and Information Security, September 2011 Vol. 2, No. 9,pp.48-54

145. Askari Yadollahpour , Mehdi Hosseinzadeh. New Squaring Schema for Residue Number System. International Journal of Computer Applications in Technology, Vol 2 (4), pp. 1127-1130.

146. Seyyed Mohammad Safi, Hadi Toofani, Mahdi Mahmoodi, Misagh Mohammadizadeh. Improved One-hot 2^n -modulo Multiplier in RNS. Journal of Automation & Systems Engineering 5-4 (2011). pp 160-164.

147. Ittee Teli. Residue number systems in optical computing. International Journal of Electronics and Communication Enginnering,vol 2, No 1, 2012, pp. 27-29.

148. Shahram Moharrami and Davar Kheirandish Taleshmekaeil, 2012. The Application of the Residue Number System in Digital Image Processing: Propose a Scheme of Filtering in Spatial Domain. Research Journal of Applied Sciences, 7: 286-292.

149. M. Roshanzadeh, S. Saqaeeyan. Error Detection & Correction in Wireless Sensor Networks By Using Residue Number Systems. International Journal of Computer Network and Information Security, 2012, 2, 29-35.

150. Hamidreza Mahyar. Reliable and High-Speed KASUMI Block Cipher by Residue Number System Code. World Applied Sciences Journal 17 (9): 2012, pp. 1149-1158.

151. A. Omondi, B. Premkumar. Residue Number Systems: Theory and Implementation. World Scientific Publishing Co. Pte. Ltd. 2007.p.296.

153. N. Szabo and R. Tanaka, Residue arithmetic and its applications to computer technology, New York: McGraw Hill, (1967), p. 236.

154. H. Siewobr, K. A. Gbolagade, “Application of Residue Number System to Advance Encryption StandardAlgorithm”, International Journal of Computational Intelligence and Information Security, Vol. 2, No. 7, July, 2011, pp.66-72.

ДОДАТОК А

Лістинг додатку «Інкриментний модуль»

```
----------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    13:25:43 08/28/2013
-- Design Name:
-- Module Name:    interconnect_matrix - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
--
================================================================================
================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
--
================================================================================
================
entity interconnect_matrix is
        generic (width : natural := 8);

    port ( data_in_bus : in  STD_LOGIC_VECTOR (7 downto 0);
          data1_out_bus : out  STD_LOGIC_VECTOR (7 downto 0);
          data2_out_bus : out  STD_LOGIC_VECTOR (7 downto 0));
end interconnect_matrix;
--
================================================================================
================
architecture Behavioral of interconnect_matrix is

        signal data2_out_int : std_logic_vector ((width-1) downto 0);
-- ----------------------------------------------------------------------------
```

```
begin

        intercom_matrix : for i in 0 to width-1 generate
                                lsb : if i=0 generate
                                                data2_out_int(i) <= data_in_bus(width-1);
                                end generate lsb;

                                others_bits : if i/=0 generate
                                                data2_out_int(i) <= data_in_bus(i-1);
                                end generate others_bits;
        end generate intercom_matrix;

-- ---------------------------------------------------------------------------
        data1_out_bus <= data_in_bus;
        data2_out_bus <= data2_out_int;

--
================================================================================
================

end Behavioral;
```

ДОДАТОК Б

Лістинг додатку «Рандомізатор модулю»

```
-- ----------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    09:22:45 09/04/2013
-- Design Name:
-- Module Name:    intercros_circuit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
--
================================================================================
================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
--
================================================================================
================
entity intercross_circuit is
        generic (width : natural := 7);

    port ( data_in_bus  : in  STD_LOGIC_VECTOR ((width-1) downto 0);
        data_out_bus : out STD_LOGIC_VECTOR ((width-1) downto 0));
end intercross_circuit;
--
================================================================================
================
architecture Behavioral of intercross_circuit is

        signal data_out_int : std_logic_vector ((width-1) downto 0);
-- ----------------------------------------------------------------------------
begin
```

```
        intercross_circuit : for i in 0 to width-1 generate
                        lsb : if i = 0 generate
                                        data_out_int(i) <= data_in_bus(i);
                                end generate lsb;

        firsts_bits : if (i>=1 and i<=(width/2)) generate
                                        data_out_int(2*i) <= data_in_bus(i);
                                end generate firsts_bits;

        others_bits : if (i > (width/2) and i <= (width-1)) generate
                                        data_out_int((2*i)- width) <= data_in_bus(i);
                                end generate others_bits;
        end generate intercross_circuit;
-- ---------------------------------------------------------------------------
        data_out_bus <= data_out_int;

--
================================================================================
================
end Behavioral;
```

ДОДАТОК В

Набори модулів та їх параметри для спецпроцесорів різної розрядності

8 Бітний процесор:

Модуль	P_i,	m_i	B	P,
3 бітні 1)	8	3	105	280
	7	3	120	
	5	1	56	
4 бітні:	13	12	924	1001
	11	4	364	
	7	5	715	
2)	15	8	616	1155
	11	2	210	
	7	2	330	
5 бітні:				
1)	25	16	176	275
	11	4	100	
2)	27	5	55	297
	11	9	243	
3)	23	16	208	299
	13	4	92	
4)	29	8	88	319
	11	8	232	
5)	19	9	153	323
	17	9	171	

128 бітний процесор:

Модул	p_i,	m_i	B_i	P,
7 бітні:				
1)	128	85	2318046817767106855733394121950655130625	3,490705796166937382751464089525 7e+39
	127	62	1704124089467323761658195067327503392000	
	125	71	1982720892222820433402831602850593301376	
	123	13	368936384960733219315195391575886192000	
	121	91	262524154918339918868085315823833067200	
	119	24	704007891663920144420463345786694272000	
	113	95	293466416491910664921583264163664408000	
	109	105	336260650089475619439361219633208904000	
	107	70	228363930590360389525796716137194832000	
	103	79	267733745531250537123655983565562817600	
	101	56	193544083747869795479289098033107699200	

			0	
	97	79	2842945957702969621003769722397213424000	
	89	69	2706277527365378420335404743564862672000	
	83	56	2355174995004198716073276976065527424000	
	79	26	1148839882282789518373899573767949408000	
	73	5	2390894380936258481336619239401159200000	
	71	13	6391433147911293799404089178004788960000	
	67	13	6773011246294057608323736293109552480000	
	61	40	2288987407322581890328828911164388480000	
	59	44	2603238220870258387136685083714075712000	
8бітні:				
o	256	31	1276386803183889055262676674452942636545	1,054048456822824510152403963419 2e+40
	255	124	5125568966510989774858748684862012783616	
	253	156	6499271117168404094220356454284421411840	
	251	194	8146828710104699401177942984196239956480	
	247	2	85348053184034373291692628616939614720	
	241	24	1049674811773767147039738386807506298880	
	239	36	1587688052118062023660524798455705134080	
	233	111	5021432562546503031198147636889771280640	
	229	180	8285097040528751608184834646963177446400	
	227	192	8915299722906709513183328677378291384320	
	223	208	9831483364087331753887893470457151672320	
	217	206	1000617429057612207794448002139889741056 0	
	211	183	9141747279553406888999522526337174229760	
	199	69	3654740880440949306558586606830406667520	
	197	12	6420599736991824427324288102045914163 20	
	193	135	7372877806791777661687799744123967494400	
	191	149	8222681678879625759827653955469184922880	
9 бітні:				

1)	512	201	9222089851822639020195625817433702331905	2,349109454792632427034905680858 7e+40
	511	100	45970830817859734384244729566707125760000	
	509	252	11630168617048003371567705924880176814080	
	505	138	64193486091363024738775640387822833551 36	
	503	277	12936447693390838614088844405524241610240	
	501	134	628304724435554381681990740988164417024 0	
	499	436	20525285015823401566878133804697556695040	
	493	278	13246499562522349182874316009710508866560	
	491	78	37317828406074405154525996559466652108 80	
	487	312	15049736137480519861086048715152465039360	
	481	229	11183909878326669974864727669784825674240	
	479	35	17164682863829255729900145893539810944 00	
	473	96	476774857632331317114906861231370985472 0	
	467	346	17404536859919717767753262646191049415680	
	463	44	22324150326323072740720485952005248716 80	
2)	463	33	22120562369381637504540323345767435842 9	3,103581932431423686 2430817300274e+39
	461	375	25246056934095095061630274376578393796 25	
	457	91	617999903394441040367878418889476410297	
	449	290	20045406690537034944554425427793707299 90	
	443	290	20316902040747468826421979722526804915 70	
	439	134	947334803976789917896521530350040211614	
	437	227	16121581205078562397647129352773755457 49	
	433	427	30605761781714039584891360247614023609 61	
	431	37	26643278770292964359859402322740768011 3	
	421	230	16955435735373573582800684035778983969 70	
	419	369	27332260932391296902713535999525053239 69	
	409	299	22688777452249283183048445899222097620 09	
	401	264	20432559355658250702448218871003137746 16	
	397	392	30644939987735971914541260407323154993 84	
	389	356	28402960615567784892096069303078249708 76	

256 бітний процесор:

Модуль	p_i,	m_i	B_i	P,
8бітні:				
1)	256	197	184187827814488709683521155345194168241467407423986572394064783546172110311398 5	2,39350679799538627 81208840491558e+78
	255	56	525632865442124045391252967657749297483776 16707136999107875610106336067230863 36	
	253	237	224213877914982825262707320019735169349 45 74856689896535039966803899069897637120	
	251	88	839157761846988017827242216437101183518 46 98271781037278082576180007210348646 40	
	247	207	200589436107305651648187448653949513213 42 07619189029047926648926134682683495680	
	241	14	139041888680229908272582475884570612923 60 91556348480054469170723149063427251 20	
	239	120	120176073539517302667157358116610344463 56 17687077048510915971826463565879142400	
	233	218	223941837752358029455087005457497573935 64 00312306451413295563809098880881082880	
	229	8	836159580085724463972361239879763386515 02 94824204209435485218821682452550656 0	
	227	10	105440828105523624586823085865895272858 99 87618250420683219226382249604424064 00	
	223	200	214664286815729711042231753287517730394 55 35330429107579289366715431930083072000	
	217	130	143389808174838809288347892345740530049 26 51364726724164138219840561558827379200	
	211	168	190573053110533125461757925394209538270 5 10322729765079395659588316157076592640	
	199	151	181617852511207702510680146443482023506 92 96763355259786869619231375361348926720	
	197	84	102058157884067232163530081283801576795 70 49944406041704955543586636317622236160	
	193	129	159980506187256388537613493440985040162 17 93130840780770295782702145863541443840	
	191	59	739355503045695237744147428796824811204 48 71293836588591808952323282173848947 20	
	181	99	130915565194222785377882608213495274417 69 49820162525083645617375043942553701120	
	179	97	129703999667906407250126118864868604082 80 97059567375029262801436354974291255040	
	173	171	236583619917463036739116284627540855870 9684935122923679798884998831381209762560	
	167	69	98893394647713564784635328977096865795 83 18002788227042672301092919493469529 60	
	163	136	199703634679369652653030816371283365871 350292948631824124809551366931213372 4160	
	157	119	181418668128312717895786752770409490811 47 34747248319357694328806605641038677760	

	151	69	109372164941511028602874834034272692635133 229181891615338584597897467254366963320	
	149	95	152606138127222615047975828637451782496933 442869599476735722261296729660435840 0	
	139	63	108482682211301680231378197911379014183932 383401869975676775652152914815013337600	
	137	115	200914804211291548893358880038627452407600 311514180461266742996137683155832256000	
	131	79	144341249650103447306526595330770986884000 189280980758864252203597619966516595200	
	127	56	105540457234442229586432682482461473116811 828821416809232348665961540922392678400	
	113	40	847259043538189832963144796161353165982040 314811626723925934296909968213898240 0	
	109	73	160299079131801099360389482191169776747338 242955988734992317248055282747624921600	
	107	91	203559923941663692812149951844093331911055 957224484990703980896615074839801766400	
	103	6	139427580465750656977915575678979962751411 389670457019131500838180815157529856 0	
	101	32	758338787483686741582854352207785407968088 614445257978642618420204565609271296 0	
	97	9	222077950329468829928741818993839219021588 192825624839750689479370473627199616 0	
9 бітні:				
1)	512	189	581269211480185502406002591348368275718513923623315417677803709417875494435851	1,574655218401349085882927654867 97e+77
	511	510	157157370133989830489294149507538827442426694690565463134098557690550213888000	
	509	468	144781658587786124202988240172691149525028305830804705473153565309702981723136 0	
	505	166	517609438127968214369437605363094101208737339793523222581297007395296495605760	
	503	58	181570581843495520837395236545606196847877559800896183368950381262555072404480	
	501	134	421165267995570414188248115274518166423721902987130756223413035260078804531200	
	499	293	924597152287766096520436478711039170936461120826576811807248372639798222617600	
	493	206	657969523307663106880087417653445579666474565100811613738511787669941768345600	
	491	39	125074447082795548573185699673964350149751685111811940064214468334417545310720	
	487	477	154232143568263555229190244635076968122599075550156195412940830776077217280	
	481	92	301181455494644731603387410079019010872904551346430490207738132950748630937600	
	479	281	923753896389935476269525409224160128179010893721859205931017831255244978355200	
	473	314	1045331371200895587668581994987465697402754527255648098585244994302862595942400	

	467	370	1247585504943253023076409491010218167885 9 2363381259273676266538314041369164800	
	463	63	2142619411647624026147396161053740512367 7 4030882028433298081712902773400396 80	
	461	52	1776183760452714804032803428486377322444 8 4783511197760564838870634823275417 60	
	457	426	1467840531813949038481678732985716616923 3 0368181409204705800565765403146634240	
	449	292	1024051946042748180574197940360665684979 2 0959731066270966053152535098641213440	
	443	55	1954989548805286675475418081666612997507 7 7551915541726824166068976642826368 00	
	439	140	5021679511986079089376079081588872980138 0 0943044187115246832868002956640768 00	
	437	335	1207115556440393464006363305220443943811 6 9721622741709878678450037356682534400	
	433	295	1072802054107154688996451866481634766236 8 4083023651356582053927038473935603200	
	431	9	3288143147473814796507273525249857001081 4 1784217365477683168157354935645952 0	
	421	349	1305355513591617175708175181827812721682 9 3953655824486111439617474721733777920	
	419	190	7140441324492991081569361680793179257983 4 8732685230945835975614049209229568 00	
	409	247	9509531514550934577336995861926759759959 9 1576268081408794268745654289162214 40	
	401	185	7264618838011211493474853270595659406517 7 8475050474823027636825344854356608 00	
	397	225	8924368366254497338127423736666803321574 8 8091924965496734543298614466394240 00	
	389	18	7286322347358427646759048274461122711907 9 2334178326585508717099331502639616 0	
2)	263	80	1940277862250726649647767671941533778202 3 9094584004187793162213314888402041976 0	6,378663472149263860 7170362215078e+78
	269	222	5264175802294188018881717625184869478351 5 2405965690915822098930604232074424851 8	
	271	83	1953612797743132473946546149022681773264 7 5977660250342035836621515253640598445 3	
	277	99	2279738930479339791375402837289788582267 8 5437982117194836216811736450781767460 7	
	281	138	3132582061055510365761391453978915789416 2 6232154441636644351840304740198563677 8	
	283	274	6175808450066778437584692313403304201626 3 5584536771633425391140278157598407635 8	
	293	287	6248042377156446170736482578746540576471 6 1565626929185103057893482128475875869 9	
	307	86	1786856868419663491927247931757883183851 0 4553798900925061308751659452193704617 8	
	311	123	2522751148149065771280371238731377660412 7469737531959288267702727183781823375 73	

	313	154	313838394476353557364352580866517576217066 564731794479289132012634503235061917381	
	317	300	603659003673431911108867781215248482256181551114660608899371051382274033600599004	
	331	273	526095204802643212681495736698376826817044 816919823137438113812454519349630356	
	337	95	179813955446344233462349685769507497953955 674251414483056424918025491419603185354	
	347	32	588234095414341335858631582386885744861355 886888291759526918342480479135143561 6	
	349	145	265016104143737323726065974819091656990500 207277797276867808886120212282277474054	
	353	205	370432297957676796443907202665466691401499 400794959581391176843947847762734040854	
	359	292	518821652887906698420438600746594805121277 211269957716078122748388933486184386 8	
	367	162	281564981604408922462168901330861676274155 214113917174472464132358631250876704466	
	373	260	444625335860270403159900648147996245822655 245602058332309974638560563233685088204	
	379	211	355118203858441866652056633967848067129899 868323714196547513030024680415879008494	
	383	359	597895610052633348824390601441592019375111 470604888783938860955269424994306779537	
	389	83	136100017529148817593705400099009450014077 452428259085524861625817643634087963674	
	397	74	118897001747870409494473723020548269494255 356608293102968609959053949490178025624	
	401	314	499476391584755324754401340038266030148099 803332095492008525317643298360403461544	
	409	75	116968156579754227274762279005096386767722 190310154658374464112579993063196754	
	419	14	213129567088519556205342499047993060004211 251380020985067256064123670020773226 6	
	421	107	162118050242273452041976930095328687804011 865679003746581613354646360672570852874	
	431	311	460270148454390037281438112503230488168521 715380827884374031661582554134882894241	
	433	346	509703824795299144528428298531569546041744 703028161700579682953485069904183070824	
	437	252	367831394732634895400616276389007702185759 903679904506584085207370067040438615 6	
	439	115	167094828997076388150901860016718932578961 600674635724937476968727471518216388 5	

512 бітний процесор:

Модуль	p_i	m_i	B_i	P
9 бітні:				

1)	512	139	1415111449163231016244865528554701902910 7 4646537197441332099022858821474980381374 1 6210146790941603735693264107536389831036 9 875332971671426913581548743396865	5,212496848716361728 901950723885Ie+155
	511	33	3356191702693540842539420232645854981990 4507002667754983839630154199659875100644 7766808824817679168675151544243808645180 4205906919553084555789711030098176O	
	509	494	5058886921936901167146490486442363258461 4956960834863735134685945973905702870489 7514833713663309062721576501277856709754 5650496560876724098090253701088307 20	
	505	339	3499082042999696289302497614647517627814 1241185509346103183650116902807563779235 4876666822168267294992066167763640730848 839 354965356892459954933004520178176	
	503	284	2943039970249397079539073569748159879549 2630123833696453248644415314605938394517 5767879775244538054550350579594183299133 462 00255183638884455354299869387776O	
	501	149	1550223613690095604004771772173366917521 1451392222344382232901471666412043974402 6652858265969837744049823767175826204774 644 912325015334364600056135075694080	
	499	353	3687397570334420221046870952968708685348 7791866534387849941968823131093989495837 7770816033564628781703588152094072598795 043 46517754888203766746550357818624O	
	493	361	3816858747234496113861266148727109929207 4524010628332593422339505657172885977571 1020904731224673267715740860029501657047 620 42403058385706625550665894230784O	
	491	172	1825966309529967856153025508163361637209 8049308637130908101171224544322074568572 9229897927734835879116190949371578284528 775 69929702731574879804312896216064O	
	487	214	2290501695329161006129399291399134003488 1696588698461069645939368317774911217447 0095772585583426593312815145101346839556 0930450782657009573230216290121062 40	
	481	217	2275726274907351274572577239118167316306 9064741739285499375232859870002527543943 7471531877976344431364773591714251280251 439 133038201500348819878970982067 20	
	479	265	2883740427786713691354941423443654408719 1553209124147789818710174801967115750957 2495365452095185833024265492274152892307 880 396587278598687367788776580467200	

	473	151	166403176354370110161563331777299517115359835020363036857457217139355497188970520316065960226516383715442169936552750789713 8394412323887227940866761441510040	
	467	98	109384302178630288957685475576172296274166368287560562849959184923886221998915715341420104660435646025009662141596582527708 14574441462550670614311139138048 0	
	463	440	495356072016241719377291213500945110380740189441705480107762448057480955973403296785714354159349331009366275330701850791058090262315803083659767411484569600	
	461	114	128899054393419791126859518985441159614061470514258029240031145402139419336364488320565840274298838440822178783795627531323 34904469312126977780439515371008 0	
	457	200	228118024013845152249538324896496504979393980218330651770334208723506746186279815408579417123312850862678123610617621758762577870949091732379622350822912000	
	449	354	410963003217281080630576961304069171297955449128742719177072690633312632344364988609901569525337487604257700239181487894087 8877818084873305223891388800563 20	
	443	159	187085101342189958215668208825667331166114049487636119069839668267225402913472350192252891494709725533123933362509122458882 2256866453030761839785426393369 60	
	439	429	509376115740163822710463977345476415611903508015311836805573264497376629955110667998840779786752327450804489387612945315155 8050037947868041938735848050611 20	
	437	234	279113103569709071982392784757225895303048437718396375732448508151880599813139483983554527555295533701980242364670792169777 4891574777342297163850681041254 40	
	433	54	650057343719823402680612792355166368346591760257270649467472009570288623831987677211653909442161031592282994741665319834034 9765241664764170513072024951296 0	
	431	82	991704736878750955382737724729850351229444317250779912348202336160637100531861926058132839591534312439443155937852060068198 2416631028148003705299274637824 0	
	421	326	403628022014616133876968155816268910983848859392961248955438611449471283376773627781484307860634146641609408690701122623087 4866552225366327004857134239795 20	
	419	286	355793340986367411567054393086180026179197517929549556654966846641763469451541103025443281346673463406369404679653274452431 8192468829139332396348508025804 80	

	409	251	319886725923669142776134384277535753015542118446259829738627458191287143457085559731930552566795197847252072432800923460664 780193679186617922608080633786880	
	401	2	259974905172885871765683327874560854801952740548072588177163923657462800516533355715014447943526116818563347855491903101632 164805545473620193235447197184 0	
	397	41	538318314351060027418085591131694586120768892613458617408155172399539786308857040467323997309762328288916876837801307165482 924630727094683828335514781977600	
	389	118	158116870989339507457694135069003477217465292766757053050990264005447619470968191847340021179842786576102577041467427647339727175691525973782821528155223040	
	383	51	694092269672413702804176206052564496286518994379717036659422898527254795478279850878297712127030367500840604046039805526302 794051985702481275699494860262400	
	379	184	253060533024752126152495760737422138663684446023733348892663099513749805716515793114467361834632335136423193379546366470327 541530506163926020286071715000320	
	373	281	392684078951554328638457949976318923323576900237230124777359357662822598190395376076723096631695259641186817690411302056233 431132086722694329677793909194240	
	367	18	255653796394807932207725103623784775198977504261033779490840490593946388845822583317571428509042440217491858967430593540542365880984636047759505920690636800	
	359	162	235215735234554484702539280576423708212318299881391246144358256387689393414415596851793434085060495832416604504017618513763 630781307034224442798901404226560	
	353	170	251026760419768128587346068855648916031857200044789069915541992574068542651729162310092562553968489851578802029870864510457006725127973607486867711550028800	
	349	145	216565055319161160656384772195792849605437565890654450423512841562509663840026245960594771857827020951509539556698806673988 570524619506642494780445233241600	
	347	22	330475304529567602408769210159852429547957792091512445662671489121921732875627847855599057601986657431320728354632502761181 406039556508397311919757229619200	
	337	202	312440463928992602147832061194290480128602016170251866283777924986673616727599924030083090687262648583638995593331678561400721269715051249004339134923832320	

	331	82	129131341871523160655770269965454686948 309669894520067136178872765059185284690302510740948229460086590332326916055099363 72 641134984082531407763866687882240	
	317	286	470275740925198566077589245120219025139065488998363609584956178999681052682005432705554368720051044691699623220109533109050 259509286879930685826506265871360	
	313	96	159872107820054545039804239454618125662235985625217290007477674203157954771317381175801856038881491901715888802636043159751306970515631828801258654557552640	
	311	185	310068140518497401880019576822737894107441544333742201831557969076026975181977925944024868499847025340596816087775994077083314245970915763891240701773683200	
	307	90	152809353871163698892891063566659629475610332351464816080030065224720398544652914217841810140524197973911972677694955672537304279856899255623410207311639040 0	
	293	95	169005870521520260834704886951900951598641461624212034925069277674591218185620105562516644955778624918142845333702458168202684612069194838929373794555916800	
	283	79	145507862561340133068287670384067373131997541834317871393363532465455195363308483669539623611359785418644316196362685483332 249980121467470213807521762536960	
	281	112	207757881514673492397515473692211336492265150951868969398804023831031482149440321563571688363694750152229841261314098791823 91077841029023541136281858576384 0	
	277	273	513722613609951896025354710332342968551158345957024376272900112176054605793257329425808242955264162210804790248813449607310 07180067652768273310585037937408 0	
	271	215	413537572868641244174878009459506717652220581667012512724985012511589582039352827730073258845961331393218432386738642544413 599422695627456918077381551756800	
	269	56	108512945549485597330300832913589935974584586277834907808669016164535030639391692117776290762450231956833801848381527435878 28797162321850585612072866654208 0	
	263	125	247742245661424036544769521097193210018020553611637613733466571736317925397171757703709110326411532424533988807158396254169 434617451841543957907353436480000	
	257	33	669308933881867459353168769993008519765416462193121509600858015906460163275347450618051787867058580517519218321347150533832380714354718368874919197035084 80	

	251	66	13706167012560951159662499911410612715514902851285436491427929090674124459104724685963291592178012365976289968811650811728679031361686264605645755668367872013	
	241	191	41310659672399381336940771297179440726423988280741551039579174433296127291207066903669443117100848160270924015767594088085288610405256639921578758014097920	
	239	90	19628649221107638309672617788688078928457059427573095621995493733886678808873825957646174531970413297033370761725319629577206752786895696553920224454699520	
	233	10	22371231110370651197004080360021223771584345270338435162884706735331026396379801639854247559088835374301266628766577927844312894211101914527357814233062400	
	229	26	59181186928657382074869309528824704632427058361445519745308101922548191226754951237658092451074307728173394426667217941869328657968207486930952882470463242705836144551974530810192254819122675495123765809245107430772817339442666721794186933 657262580154577321980954689484800	
	227	182	41791824954465983905733701839077533711797609935417254692391580344425000236338408296901287312366570611569141526849141128113036678057973030284014778430275072000	
	223	43	10051002892143657145416317539329714841592087679978510398966003443106796119521490252452901627556548951350869522583513195474088380587490138190564930080764672000	
	211	205	50642741895111571299758289023526718172972808143967552395980474749435378001093896934959153301416029462251184858911639560354673241326695874284324454826645760000	
	199	175	45838540126902678520494541541701778858610637105429632852594166690106663382531980495731504734766194089805736019495337437580495390022999023359447983686086400000	
	197	179	47362281011179124338753765460680465981800420720304828189768145371088889234203674466670106748574520657664107984059272367340241516693524241075762048640524544 0	

1024 бітний процесор:

Модуль	p_i	m_i	B_i	P
10 бітні:				
1)	1024	979	48585566090408393113363049453357394911360968191609477967810125772557614343324456200731212186911406279415963422310698011604878701669415335909685247534181321268860201590945157619780012113861383960126814861667783414729005196629985638425951731495464996766272249407892831710762248156040	5,0818814787107451019493118120774e+308

			27768990176677426441803366 5	
	1023	142	70540290320325103076911268554740379636802726891 36131460917123491998695565261620648711828258206 71417886797369321874114615272851309797671448807 70995134882758486000978445060378198043288346102 52366034845035486555945399743776322891125874289 49221360453616842261813019936843148794976008743 02669184083847914953512960	
	1021	711	35389008142637999681547117516033769260989208186 26610632004285030745697545543402328359723565057 33689571267969105387636318827547018757684209974 08937402120524639657522778655282216745099328470 23002298590612004423257062490402215298332950326 30604491723123954056128681829553099560960798504 136164068994351246 949155840	
	1019	451	22491938634921943483602940404778388902521516893 14851884972643724774298310295546381814622412869 29075788666991026925752866240225315730197424532 99946328467748642663844302017195725141899731928 81701802785497074969091623184830000014683974898 45463729102383283365948014794483084831154864671 105650316302348090 960808960	
	1013	966	48460982314260412324610416687727450526202965089 83725345191800914183200738367217881782396102546 14147936859643449608465857 01745984163470483204401630789538058004407961863 46088339113270005880054010037438193103071336811 76732243457490192034804410500553567244786749205 96740514258861306630533328711830381 5680	
	1009	777	39134012972827046027895096907672524953087717812 12974741740291508262604965629803365947472493634 07439109351408974334789261684659608223303579185 38311306592354109162665404042203184478333485220 94046847724918132744881193092962792300836216880 72346439791961691973289393326413301781021074446 58521864502514186411740 6720	
	1007	567	28613970192939349283071100272570992170937725015 58954102075492137727747579877418425038433483026 19739221558829670356227055371190362243416608275 78282158015392334628371496881873946019572160763 14665403595639837096239790209924725587151589051 81430904190998480217620519670186467173578387266 118819923949111350 389181440	
	1003	492	24928072657285010868983663126042793392552624723 05085547851489845603201281455083944242654116248 13626279690017790016143029854570394339604287790 22182879045013178871883270648624066621512727456 31963169080128544613618450856893069897539712232 23994423952138383848346988571280627328501920166 616504311406246030 659727360	

	997	702	357821544438810738371957561893515650293495494272419813251853346467917500770647987793648877358704208449102224694524847722392343987403024962405146192606944166836337594942618915630073285720542917178186660181567816545767188466185808869009669972963343780016671812601901171231959623123437050338790124451133203875840	
	995	318	162415910575881099740691573491519653628472236506025300451326570293443530973000131623967704578544767243559770262280526379752899958238742791207750976794601526605668641106068170947117660874966355595177265048784306095390490965440229579069191865664699984365081434949938236387647454756393395842047923086563 841652736	
	991	418	214351812119181781293119307512448321499209514197656210011839999780242620588378693600024141840281255019289784316030100231019517232549154701404513923292788305220454672309987411796151864689460671057093007845578383908426971985086679119086456056865898020858420817578422189313844896228736544830093964202770 746091520	
	989	908	466567076104080541210310932797401435498596278541101862454787261121755958603972944264233176191350310693943154705407165496726745764002691754407141123092542162616318466953114287743425866589392447825510972524736469672502855204852846136132209279920095413938469304550912868747365579469164512173075581687197437153280	
	983	551	284854190719188255460231007981145487448709064483789966685021259302894325029566969174655221138807129751196204173605338506393688418858325948564890670559116022552975438246403783760732841917618155253389251478501672761688835100314717062953188782866412800399988711568772556121825047291115303887546029650579324513280	
	977	105	546159217261646095910622047357348442248271145431947151292698848078365710022472488345597353464380904120910082789432352713823095444821866872386582700881790440899404262810569905296669079430876987246665133907415253458938220108831869481255014093539518450023328931673762829212587629651737019513874 37337600	
	973	643	335832455376259928114430369492886235259852850333234426526440205598941125687786878457471794662763365299956975499664354456715627281835256506405175456471052175077429306851163246489821639172404447809279756311079871225088086713349031657406526265050353505326241300844325837122014604514725089229332320037028 859171840	

	971	49	256449219831953151385701625944174802371158693004400038976365768761273126731458245643360130473251012927078891602947329914853151372958238760058485347842812552372881960409302271357633042378633991639393595316164974307779809022017060578393488182787713105782132968054208483605689254307235619580379641363226010265606	
	967	634	3331864382112318918961596369035249494268376583061666189549831028787122796445265045340082948232826011590846389581193292020772023572404265656532183270216966436520438459311171113014283409135935643857699277772546239143805920586032988056702042576786347879836570088419313891957944377943765799363391580394194156308480	
	953	548	2922215162994216490942521377773796298265681089088919972153158371716031456385266380555433374860497462076837802279827351357318043359574758766884393461463119721139034074768679208676518896588863867555826867175491191060755479347441036260312272950442307756205306751298558104256292646870559801779057545591867 812579840	
	949	653	3496805696099174448443520140449480822258962051621446891754401162084237383168719158886171359933355347402903663285511985716469188167824772433258712610680571501585055256080588075177628002617659935073413931462839769134293650629603739307066736148532843056956550958546173065965767309387082829573434045325717161932 80	
	947	525	2817304938039219829486154911658550306634429298241881556562654564797060711150768854876708629011088401933100057472415040134134974918642893000642509497156859877289957575321683196804891584133959939830870203484645384517131481675726721469528703712704058972482098868388555260006685766318363855172898986615920 07808000	
	941	235	1269120241760919340019222397277571130995312116142380509094146976616085979379682920517255446269465448931130514225390628220537559826465022261683087447082178799436250409594583243829653555166422984281485773866687732599291585627295386325461799236361975211843588265098995051542306328354476563229498112396012213504 00	
	937	759	4116486704740080610863956953433046052854700716542586638319274095650511192836248578800216478935177199619169848268781167133877315296432806079688607976419022502575259798316190875472973194293273949955895855606584371531294702400786954862667184944232219504655944808513995080349210851885855248816771099876779 43239680	
	929	887	4852130109382595161925769189787593446141878377776205539628904905898443240963295257858489116970413933366923541291089111993914405550127479724674152257947190673975937615945225991698075917504948230864919434834924992352577284661052874141099	

			8950288186371259137131505640148532426182175693190554981357258055668470272 0	
	919	877	4849630094482397665298744786933514777171292992064975860873328786985211803231722676995386649790476456780302247021985940654254925611599309502048015638259429210688591689721931720663664964119078242952265900881629514697722205357464906965441752378405642476329053878595149025308031827466627764411297773905937773619 20	
	911	234	1305335088933385679315190959413959397521459962640331334841192046939311190151247873509944861787881799574661244128947212510886084725359736141668242788251457498484553904712005338028478104147022643346696305162909305145005074646129716218085038284099720997719069876195007199847287757973167724799757607309841 94754560	
	907	285	1596842581513299177569519147124658864932218817554455027476196049279997271246992723446105476824586977980225841364030991246920478932466264727061290703491679942418592113805734627818882163279960346428629628958866731882713710814029009273940146980711423313651611432947401946552222302515870872456322932167995 747251200	
	887	419	2400573099864489512645729029605907879573511793712977738708681790367788931679061633613773226029305417444012607729966137370363062442461092913306368756977209215202693157641309154903749066072403213411307200625506836438945943004362770155277184586406789438703150171867718918309621227840795724884565646848741 0874880	
	883	167	9611259421797218935736524038696825161924787986575102761987321507158288397946026934797963500393132064440874529239173082719588246743427813949384937377249856716465070389463385124122241258463067044692062596362900995869057362789937017174879899982967895646639086532638335238414438812651306160421480921842660722 1760	
	881	190	1095978979517640827889181889097287620502097970126288845948475078847489614090822852944092067710859166877082737886625886538749053644908741662841158280792246468235840368688461072593057981153934191608601644695945611666758483320714575415409544692777344642816505160562462024610568012396439562102069541455 71 050137600	
	877	64	3708556609321410336656282280193330444361176975790714174515922633723630462755623581611935854636565040220694538378414090295914699676800965607491800306318791626058310229138221755811497401991443645181503638872723841338205024518704352170824687508705383204738992196672499477623613203562434631792779753072002695 1680	

	863	443	260865990158616463518371394293197965720814967417697377363730377018866127374646957074818006014757119123186242721599079634080988536742037215587385361531036365409824045016063092513774744196316947065670473682300262949403855062425106200434398567622870280680828251253155678691886513131743380872038791759443228400640	
	859	760	449619315927842407157331429240843023824143567721192304205171848411938696164849794385911577021311723408013919477586685001461194999378161306152763886089857573232025781054497883564369770149762990829885005344413543133138171737159274012095836495616689466971520859816311545369923615465719460823856884854393255936 00	
	857	646	383068312164193854826050808705019292701515082814035660344698045645247420075571852241392459699079060497504507942310289444360433522515711175831319196532683205478837367604365962292327406855133544637620229465838446346162002443275197106104754979316832922181162142117712806128179162699217627108580992354309 829150720	
	853	274	163239803653780088855112712369192724047315010428003559991908553824465297651533344705118874926263427981197328579875466931038927041129820932288442378005530662030803017189598551340487837540984013234507796273050592325961311360967837488632918881698213323075243688979339148496849173063754014718372909785993 571153920	
	841	425	256813273299889021204335020229834105580135197737365104915362526689699792700391981171282103043143152611522438283497025825940994061577667937361099373731545706157584661146615581141282767753652800358627367679629160553430155277150356655557117949554513591517973555756680545443902577158481058289046491056733 465472000	
	839	313	189586281625323386997632252345677406195542953112247994565478858551804656893787512808413662848925756128132611643216076840047467404644953415565797578196665416445290564699171448560788857334634474084842060930833128570317829528346406319882961029839678261897358852983238561365308300057935762228424805385506 892846080	
	829	577	353708759133425805045205418042059486472716002691947998909853327506886120219723583643604170798003311734363283429035453803217622806610358079273497477609731568105899552725532497148585686601388765894064970008128312105799636299791813296262486013899026756203043479020278241511308686253297503447577756647178 915077120	

	827	342	2101576137507950211446994727606382960812850013332852299764318960140688065326755598789288030200580975615088035962045140112633916892498527846352489482080223034254408291011078076079892801105089287851173504424220738792195408524775300717964275663611019509768336014050728269198837671170826913641429115889317669990 40	
	823	708	4371776533325890075552992421568427183685762004670540481439338961955387553942076951321682797534776386910830744972226919976796346568424491876407538500528248065928048591729169387400234889229418329267008113294152864270111085593969911020086159001282107605471058776558895970626031697872062313628200245212742 30231040	
	821	378	2339770035265117720507722125414453034196856605010042046926529096785904842011011249706619989774568228957448164661378161955004374297841011161459304111499918804788531879606987175285563066657611748324336832574872848939294676572201592894985933422009909342018507055325277123608962999886130585884396400772471310643 20	
	811	106	6642163215084327753472590037980355748472251349095568185329631625235360185702224865016352899944596934128317772748422677675878974864264244852104900201834299998875474636273646941733747331302569823970463572689699324492952672035060765769578541156447862081514838790595501241500584360156655522516961918254460 4375040	
	809	807	5069318112879568970671315985595155082203598253378291600660005734063507022713616884321789767121113846185016478479218962167684590 61C6293236211384410865917708332719453477674133328883524408761247180104917411003992172915248104292399944009334315372443587838743232352456700165981504656135543956087572420801 6477803520	
	797	118	7523990144138869787076772820892546156782480405361469711887444512106407270795335280087295989901967725695553540594192294313892499426627680498292355052142927976283529884970946059871169623245109337069735836974253047457761236812708567063009010577279746046220580035775766172783822851001060819983289004203443987456 0	
	787	132	8523613153619038798695160853802031161869188600409709253864780570409433705734633265202591770095046762152725590470908687043683872565620770778781781501363734787941038464222464726408874389376935398284458133149261490801222332798270541310025197044904899565466262163587897663747219641383451451613313122113789 136640	

	773	344	226153587150905086037589037950146643575847905779501329617027746507684069180105628858615318586979046438593333389203974113021333109829256371670816321985574886970972462015386168069330183158139251349283372016967800464182562736495145936513094932742378816848514468446258534939587282010720769720852478462132820787 20	
	769	381	251781123977736525857306606034004968849055305420956977838553893259206906683553274684872279474296556575955423230280426871181333307442141629793255786523171748528170839774373781457571990289583673047385419656618848783571158216353689068860299213689916008591082703921399331344931008206154950754518953901616881817 60	
	761	22	146913787820809976666077345421423533166689694010565947936740742639339122348076136218012258499010113924971420054765771719303092591089433784557627289565774403813175586319384137969421466150671225546427256916085606510305781338419599562220539423441527725756065447278661311024289572005448451432756842468321 32884480	
	757	148	993551464794174735916113802097039131349701384473521171143223031222738481555129042269862174266032851635744031339507585997458190966087471375474747590321486703054541847963227854642728527057631727563528827425536789362478656213733797255540705787400767665212810087432512677917838712371180988829622772518439 69167360	
	751	166	112329204456189572160264415553242641634276977881483803044768860033029620928114429148266653959241929861311824293901106876968178638554435372642699582264169091732860544228195863776329355604343864171274480433249184892996538756550020180953104123021876476721103520717371799881699579186298759688260748587659 848611840	
	743	513	350875531437229103270524489851375243995738955792651854354998340518778404446546476486959865607512880249060122289490847417190090700557795934508783741927391207805194277927219697628143609659604745838409250637718146227687537587480990651498611569409080713575722856720044201607154526391306969902044793843970 197795840	
	739	545	374780163179615166517236121459025115409144526647879038485813379911137312596843628464383652055430862958618966727849858480270396953446839801798210912848795086169857968680237263514132771427359684089389670744629873736725127821397692338744972463865521076508176675677133629233417262525524160348620596810948 022707200	

	733	593	411126291524620988466022087934776522899240718623528010810224514157773060643577838394586667057074235032020511238833648152580084973699008137533635990600370287312310469831994531153880275915985712303192778806230310720111750353048666459394452978015183811144220033422940132766118244001083585659273710599074 606423040	
	727	158	110445292109532011844290408020389663504416877756924024309227419116348989576641105830820044625509429024075988524102331249128205870747851315416030949373574982904142841989934412787090773083808970971935228942580349853608519048198673328512399769887222775349867433099235205170304673995192640386218689663036263208960	
	719	532	376016821512394491548961597221862183901888076871700879719846776703892542425485648628996705511703392608120681198710533645867338738701096513754738421009501194415044648328330427520522312644022929605020658850620116113986067685054063231534390104053289814311509726018843021183497798745415453656938128631842 373406720	
	709	75	537575614814253713182226214253606113315092594284719289555678797675360969868986118389732635839915518160516976511460214186384408875267946740199165120654351461574368290112761230581146953490803961922191030163199512217957588786402015964465783279454109177300130491308475081850963413399857056387010949088308747520000	
	701	307	22255886076522093385141779262592994282929719324735112643377596602646142612381003421220042741763867128744195275019299020532760739708501586554820899221490842069550494556502890676554291837882807943674724906973370504592924897387469710104330154247421733877066724323286088776210413590271628428790482969359 6855751680	
	691	165	121347386973556142087067503472181594146003101094834319515043095313173377829470990949971775019261971812847232990242495237036526912249340568069849168624697164944095464387537621919026485970326486223767520817446975276059370041827766208592435421026413819790556228703756791992928036500882349718156248681323297305600	
	683	302	22470398339248096936876898495569279408648315491079653458483715529654063474164484079526163498442368163493133035955263668357297581172205411443233740810984563445462291760344403937447344731584960999288569838259434486497734113375661660181451358561279638618867611622838080541876931996164098233181846831994 2 396098560	

	677	22	165142381878340313504999793006947280265658577757814898640856285300645601339565642041222051281752875475484860652402883719925632883041446248225043378669947298821014211505245685368877009956662928568435956444817055471702658195771514426661492616305764548449580214740120026129223580939654758552921650101022 94128640	
	673	73	551229343158817819379345857773628362691416491379776449123987469663866423328560961456114750533264140137372504275456093227235212393-35723319960142429910000368433968887397051669364 8023335-52511-22923534085816090570354597508235326228880209122375355793725337499522268993523796000671091907214942550904797 14 62794240	
	661	552	424387076588249817893497748905709162593850042239076185405601411373899489774453573740701729144520537592426037444956452053851524624701695975427668895492188044000882236943723927967891721419644911688480199884843301457826778836422761843632969582477853354516585920569672770214519110792688363981990319965291 412807680	
	659	480	370152216962239400445473394506398042455017322583663467114999713555647638372915236844499363943429537967399367620394820620961744114175392280474617853032046311386927701550176837615328135987840979911704706899625812843733589510952623314166910198611679181019282567126151218726393260188648160707390118753127 431372800	
	653	37	28794734259157361220846024050055843125257425269772416790023730269357313573400945061190109721262896963562719438131977128639963650893116992007441957345840943116856362132778081392210776990146524416753113413519576935198941146776281949557592430362265739761336035076049467732917454259762021766616555849787857832960	
	647	298	234065020194096142253615907264153337111673144782409193067290938135900291470085780660730532231899083993797728657999664036533066823254739460407542514213483123141382135129529970772197643017909174298698922925113227005348073652109031159284072598652717057152522832862178923104281337659940207883780470793942272921 60	
	643	447	353281651785956930104407835147528481597012110437524295290522325755662881604538491650449426953433998469643383612113023246431324031055559325545132208466851276840919692291149046095321588265766851268567232457048582707416843775952070721630667567183897652249848119641173350680047439844319124689146025957312748968960	

	641	254	201372526613499103883794883037077296978599615048066475255172068555725544711027008042295146038204942284877504382824387169982782437257417486542915964468350571625754941015593799210476194001757508656723232153863645048950827528212154855905949137106131472289695891123823283598986161758765798973502347951153757890560	
	631	126	101476555676315987772680394345761539412341218843805838379645028444861483110989975489124934580291627890864497791177573743531885435738376659458958725596483536118932101064835524083964462637395628387442183811091949761181242972307316839238428491255685978330543808368835315292285188743080465452249838617760113367040	
	619	550	451540357559113054292749837906717725120560812366884839983278292199261195908263084256491634562789566627234453399340679638084222382144826777255066104036972218504954043574520714841396347902507280455699283171006246180705571884237945342689137727136520998789845740626257098907448967270380741277576563483006 992128000	
	617	590	485949768628742238273921226762670930871607526991336414941190703091475412870654960958636540028514447113400013926545482866140382661509097354711426821439052965729171257562548136930090939199866462045751982339422988559468047466752733221168410164513124602823160639263637410025016759501446148584009832767434 753484800	
	613	382	316684946960441211899614537065836155558527565091647205969942216867426676144375121984640364601655332479734849123242765611336553629058880172315522529522095141163412200016443776330911753321462454462386893371050852615741593108714625774407795131790738491270500798270809731850682390339076640430114782880216127375360	
	607	129	108000446582155867899746494853045730323803808381932368900718151596235452312328038163741541145644180574363631386523146951567060780100783474586174280285925621146739809295998363339225220237720859223208157736971753074061766124092509131523349817731251574882036759726665435389722053273946900020235932816179695457280	
	601	417	352603090952143212564536277144140672700936374438291984725468893109514136050305829972888791977767149618882586037809754198218767913920342224824870497042001325051324838175642417327788761007302643095769616115549733862623586244584828669094221236588007929139504266255535126649913008555054451090846133806057 8 471726080	

	599	185	156952933816608988958367727084194203345518353098175193354636860316279847774881612061411866844613286454144038339734399540917331085419076759439562588871737361062664478069316253331499477250130888515357120693057460256134462177336495100677027187199996060635663029254259619612646891749787981749587737645337822745600	
	593	396	339363417465338121479245780368070751655533799082079655115931651378358734894426123762956478401254728590937354453656246376224917759315862391377804150501176691978563099730678570304407292298801761810280027864509382458375816656605678718937090897521424416088800083781298927219345456548167434692395148082810381537280	
	587	546	472692893931186852753717930050131690970718352172481142662253360779783501396621674581085654947264181481409128552543803190341883151256049822567767419686515869837365147663171950062896358327402805032474863033723602436666516809674571284652373034509024463512464997188845026316079161419357784273029770687953 554769920	
	577	343	302094514245716736563018015865261028027169087755789785948415620497465068005844314668616777612944044170772136260872270612002923723049765832256070174919860817760568229619068897472761141949319799163424309768452916875213060991793754029695067448597588558059126132385520808157343130247188995890440281169124 296668160	
	571	63	560697956495231070793006382068086964703916892210520875810472968027211872530951615881241976533835527137788950142145788372754988703598210789129623081010342480657146723046508018362360559756538016746725201267942384407227358279561444181781509439425024976589957470058976216720069650585672274083797269420373 30590720	
	569	13	116106255225377304613956157393684535358528610228992564542113449986497997874222496852915423656391361004932278207524986389124899744659511777490701931272903513095042824369521271523525049099583834025467714704492323470718572837592674426349879894240855870818401938802348326407037604717199355330265493093074 20564480	
	563	316	285235265945398836983300627462960161164160107031203430454026052211672168462586443833059227150072308705162499669706553527943892248787523645852413855042944982491338707377201840483889847360316596435513006896113372446086654877585916553564883242301992744864488894718095893992225570140337653857126065310932019015680	

	557	33	3010809493670638929341603048448024472347868684256032846854390623389687758713265879584577972650270108534198850853054370154119931646832830611715108636253707036853499672954703653358969544182965959806942796583693354246212736046785868945686638273940824038609492065863409093971751282661030651893930622577895814656	
	547	383	3558246081071691725862132402240682004932929889097688456330917988357129683973517940045745961350921126384738657422485966646438198095122123911294642072354105456091041708110907346267366527228941948765026390438180083954898948499234813628343450388444240944992529787284399343117210530615258600627024063183009761433 60	
	541	166	1559320379789248959193319336053331309932384665230948911028122529538862350649526116515389486763528526034229213441724817856395962579368015684781402269942881310790807525947337097231672099604850645587270758848454316939075897628293004786650868094164368579953557127309837654557557110704319505708654748490304308224 0	
	523	373	3624362890170378437910312248384089532414772267858610813461469669772668859366283286449803025527508940374145011020809021366186401986659920741181659918747936889064951303973244021817676988637795879109161842291121820334592990635704727493045956628033023599777198453536402669752969570872810645275710760052182700902 40	
	521	88	8583600194367477331507474845735378790007743350636521027244507113131445075384717056576378404433527577497370492259251614463697002826799163919259452388449468046205498747719871707848732110607355289123312285078015090544545074743747813193844952110480046011544399645225432452167705512180136010773739510049330614272 0	
	509	453	4522774675551999078945065325876370678616935013872515710848048440104241263120140263158065717908488444695228839948166528139892214520810303344517282408812118909602380728580126097036307402261059438854840825009192658431616620658814414706463048496273278932225907290294226623802144760328117610981297068640097933260 80	
	503	354	3576513008873963749682020639116118859019266530659895443563196784979407511823388996956008889036899569013532508063760193977041146347726994845212131397549451449561415498005075950129615620115120561617045404812214520173261852329858088240090366658106535347219365182015876127062847005316806445443348709552770293043 20	

	499	474	48272781982142147862203883745985943106840322723 01629198766032086027197319973031098537445237038 79889181382226394924217343384002012226580599042 81262642037482507802713102213818494718432083324 12138071958603682720301216837275433292716021754 38673595256258593308359898661287527924837642314 927219124172655475	
	479	86	91240- 52755373502874246516876546636473972041319182947 70039793739584535951786098700819918646542243230 15152738366312459284637521134602304371242975478 54637233174768356635589431519707318976056453094 62401078126910802499024639850264863303364417447 85938844166963300683172154231889858556517304276 732236874336 46684160	
	467	105	11426071847208313398387103645998488823909227175 31075731933547697157523123537378203669483114207 06668806451638305198160300086758939955238529833 33860787184352478773403956443828639718339743789 55059109136652119896864727076451026515291319761 74835800031856400023453034859179067508840287087 390614506866823352 304665600	
	463	359	39403789435359773036712806491053885566077417743 95504261768077179297546523303755383725184666176 22856019912301075416788414901684594473220307356 75650777616337470406201656066588561857426988903 62080115686251935761988318184680529638344865412 24719662300541417939565394492357260208980780645 11567816352449934143647232 0	
	461	192	21165319824565359209853966766135905421072140180 70578957299651290526436387746213805470933456301 25870299797122136086929968949058022769269859290 90947084435446351755779203606640866681235715053 11980579725829473782305110806381672171616173848 10256733272520860483480899031369319473456368172 787771080336883567 008481280	
	457	296	32915468658607889500590728585884403607515271249 29783486019342821162420264933184617060331141879 15478638187409944860937680558359107802282639489 88355618090390993973224459875863718360853172106 22397223594408582631726684447889513332536203257 39291493383423442493319994053434216106741910277 4654459625552719732370227 0	
	449	9	10186399400533787509475235258061649915556364793 26145148227108064726496911279584386472125894371 04704421150967397761770652511326296453890665158 45204557967087383239922967214638195510909310244 45458359465764950668474376477319687017450295250 53098921950130968140539422104810609977655260712 334871730270709205 58627840	

	443	291	33382110842095413649373583235090973572026061387 57832930077895977654862300749416010836756890710 47493728697372691852400092462437486038848768972 61564643577167554721472158848693553731123641014 05044990956543333067273477706298294106171008957 59038175146279462409700033930084010260106814883 69606233773364921225727488O	
	439	85	98396338426062262794007176315849878265555216991 85606069877899086379841715764449028020421351220 20006133441323901690199178205724399939948680276 46730436551225001607608211527883252764832556410 93626496839007679641185669009593990134393955514 17747495881232194149250074571019217868188566207 60829652892473217696793600	
	433	281	32979415600871117174313085893620228591117038480 90849556239825440556258373087859442052903565 9 96936990534307520043162648564716840517652279 01 78026010107111866770746672847074767423387629 37 55255989314230013082014089667964503329718513 43 67499724002536188369032415632113816230395245 26 21475018163796090646932921773568O	
	431	193	22756453025317257649100166583084516543757873263 46113949727159153519052406309743340513405868226 65181000868249079346268611360505358901902639098 63312487232634477169940824179399384078083735443 50865817381407407323485023005802838821118593336 56261749920920322406669881447191880093675481504 22281463083446371818742272O	
	421	9	10863879645699930146684989622018244209227571952 90829386113946605848449199915756269657920490671 25919917094499671247114068830369375552961778280 62937877736869917042103117053141448418998290498 24253689786528415321009489402177053374905421775 50692199419498348444423279156912028218449434821 46878243917232406961576960	
	419	333	40388222730565110237449184568538947505283505563 35381705846526558305797896679650456434708229877 37578651212654267068328365195862473266941675483 85771675777380118798291230151977169933827296203 13221891600179743545433130409955121818830967779 73062622997022857689446639481436435216894235373 27427781741940143971475968O	
	409	92	11431127042576737148639038794892981111431739116 54561363297076336259770019319127189252664396758 46811470494543908360315732516649460850782819787 15938872059343809477716325175285916331557777440 31322844683830985661867414928691160861636424781 55213536210703885681863690527180567154495612863 49168382053850735844493312O	

	401	397	503118939413507682163061543489959346743604514683633558068513975787454291843501678143019831114774864981530157704439703633197601726843872893779150803012276870968319753027129648729400420415351832332917292878225291113965673777140264803212227488005940058062176657622844245100037322892428812077875753525298514017280	
	397	285	364820206909965328477469487768782264607940168141534183859171238462709704035522015155067422539017730233769480634049397748351857529407280127819796138052129397424096485446297558547034287681341066551830497094632777284035600934086728315230154486525254646217131377753978228091401921506774529299215339806982727347200	
	389	131	171137910979719179525799446627800110363641765431261158165010766235783552769362679314207292147988049395820587014251630620110359938049739932418023161066856930247012004858211312202729555761002163425436780430258123377178004760556068006044961388902890167192660907976828987582641408305622360034855835271201467560960	
	383	374	49624638982710670186136882969111129205677926146597693693410311593212277436476787839488892642341875818468805776723352533355242519823499966272068993501938240765287874729470550C89C453C224314194383658731663796331368768300082901680 4C555135379589459645399755140186537967051491473088862889337299152143535222013747 20	
	379	281	37678329697037450492025240611972451134442421272383584539890353977859099618655491778599135727616720762568737630044575905765762320425948200915792249014193768632541713305497029800334679471246924372839471151586306286603860142946732537760616899554041413292639471469796370125623778324798635527503185356522 3 090447360	
	373	341	46459023706175980690743038281994672532645798544279641793239287662260713610100055355484278689680262702466498264235647729625727309871190782995964951583729007511482912290356450386549890187137867049138946620742299506496296079821136370942959055474652554669052814567706420755683851115020568602549151081369 5 505198080	
	367	69	955449106351611476933249359763875192351958031062770294799395548086333660561695545785435432440490719852158074765382203023146654553568872751218287261309047333394356807288152303769674481806489395865413529032293672560025547720965414096190030292976366033158934923353384540571516117963102890300053138613956 48455680	

ДОДАТОК Г

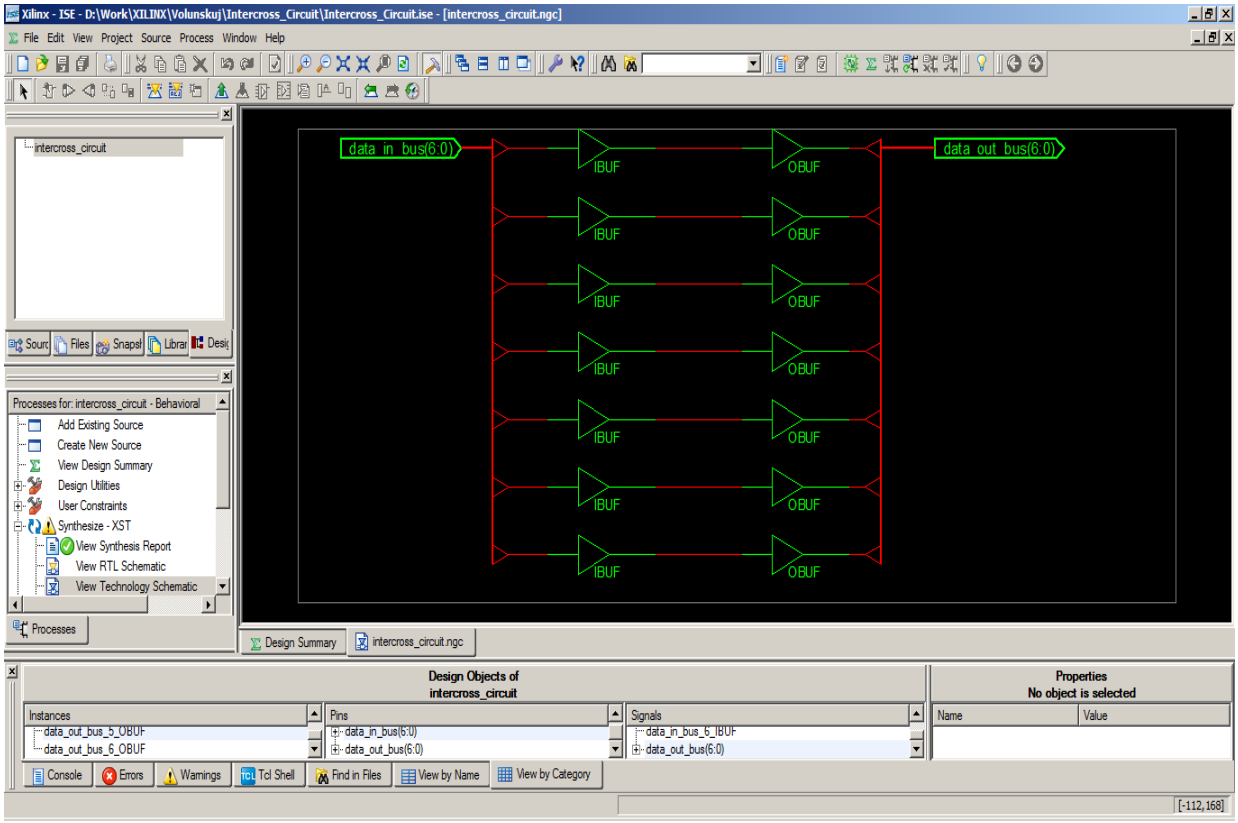
Лістинг додатку «Мультиплексор»

```
----------------------------------------------------------------------------------
-- Company:
-- Engineer:
--
-- Create Date:    11:16:04 08/28/2013
-- Design Name:
-- Module Name:    logic_circuit - Behavioral
-- Project Name:
-- Target Devices:
-- Tool versions:
-- Description:
--
-- Dependencies:
--
-- Revision:
-- Revision 0.01 - File Created
-- Additional Comments:
--
--
================================================================================
================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
--
================================================================================
================
entity logic_circuit is
        generic (width : natural := 8);

   port ( data1_in_bus : in  STD_LOGIC_VECTOR ((width-1) downto 0);
          data2_in_bus : in  STD_LOGIC_VECTOR ((width-1) downto 0);
          in_bit                : in  STD_LOGIC;
          data_out_bus : out STD_LOGIC_VECTOR ((width-1) downto 0));
end logic_circuit;
--
================================================================================
================
architecture Behavioral of logic_circuit is

        signal data1_in_int : std_logic_vector ((width-1) downto 0);
```

```
        signal data2_in_int : std_logic_vector ((width-1) downto 0);
        signal data_out_int : std_logic_vector ((width-1) downto 0);
        signal in_bit_inv   : std_logic;
-- ----------------------------------------------------------------------------
begin

        in_bit_inv <= not (in_bit);

and_array : for i in 0 to width-1 generate
                                data1_in_int(i) <= data1_in_bus(i) and in_bit;
                                data2_in_int(i) <= data2_in_bus(i) and in_bit_inv;
        end generate and_array;

-- ----------------------------------------------------------------------------
        data_out_int <= data1_in_int or data2_in_int;
        data_out_bus <= data_out_int;

--
================================================================================
================
end Behavioral;
```

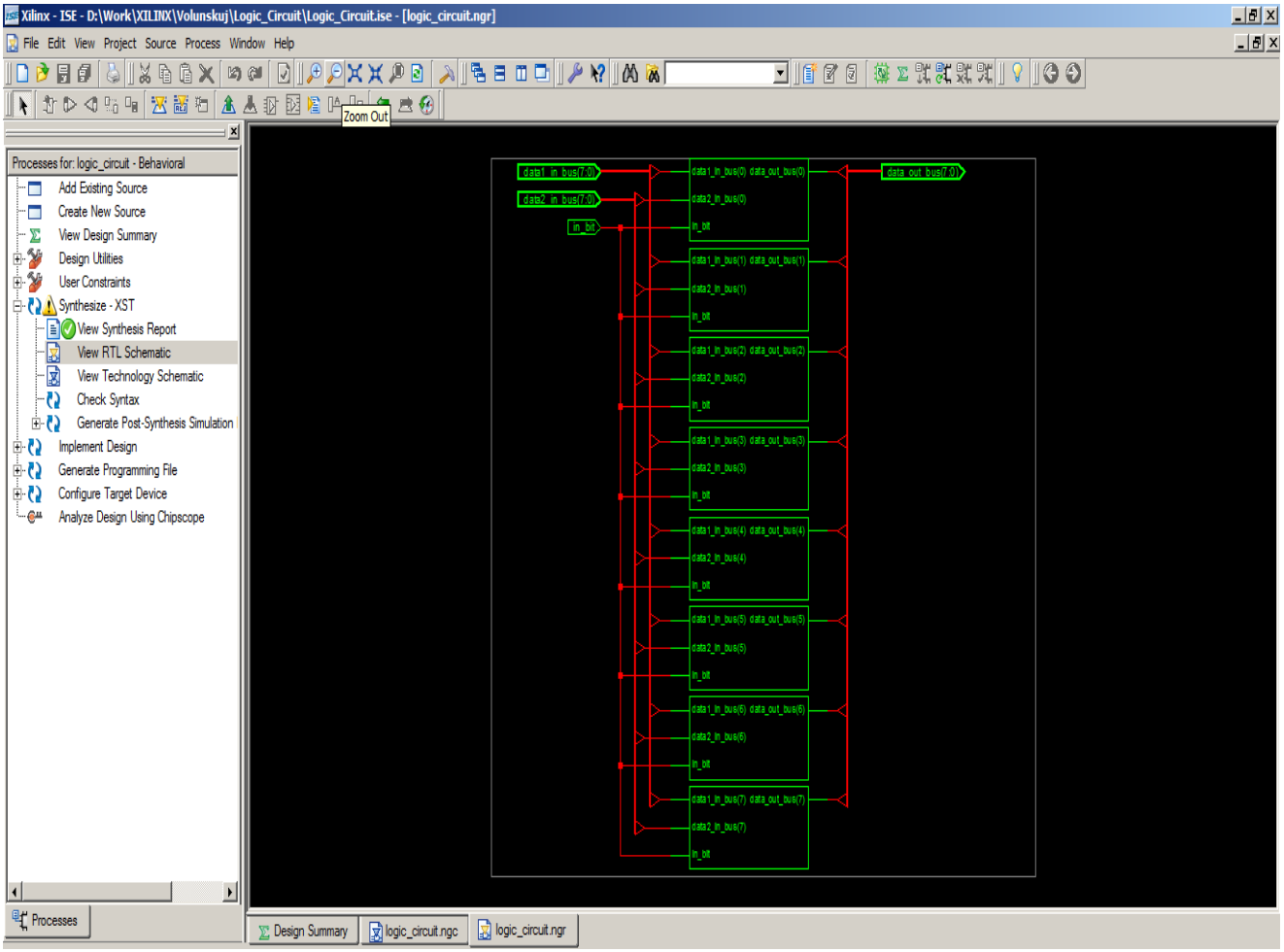
ДОДАТОК Д

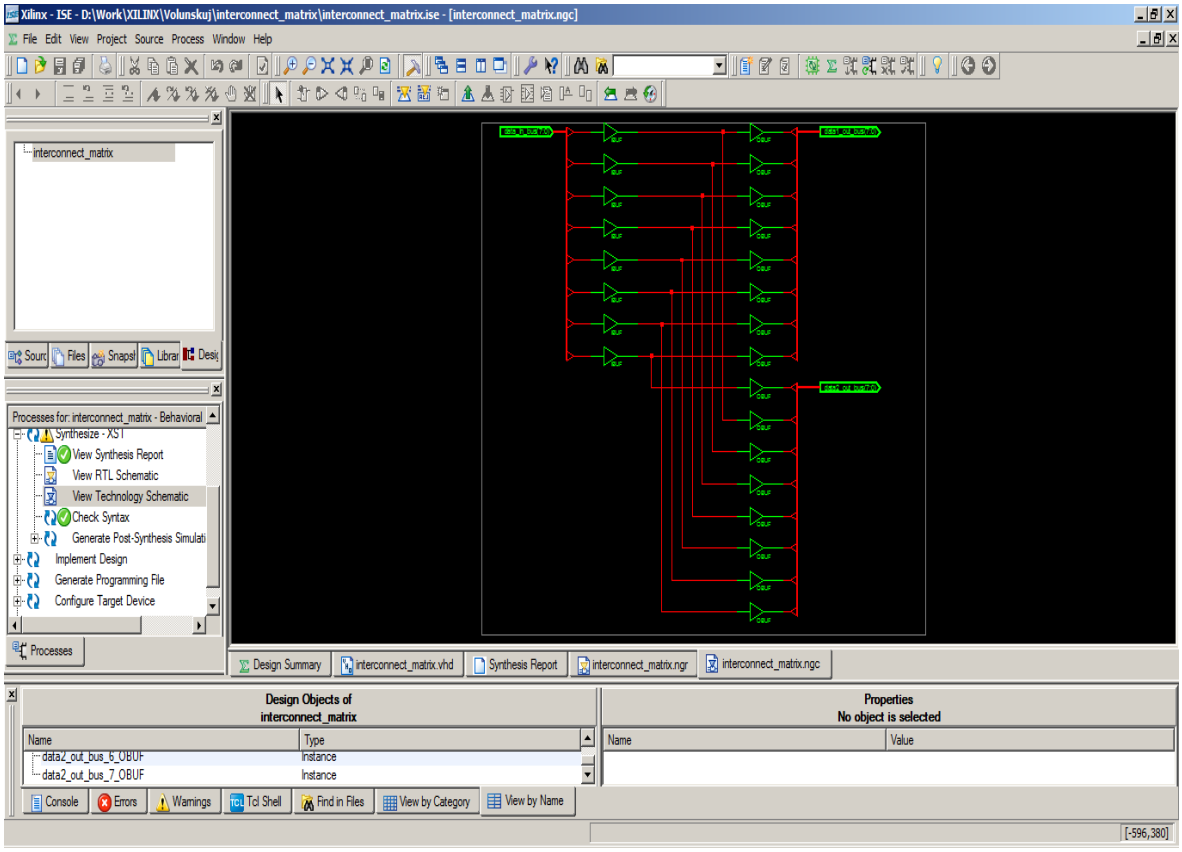
Технологічна схема рандомізатора



ДОДАТОК Е

Технологічна схема мультиплексора та інкрементного модуля




Xilinx - ISE - D:\Work\XILINX\Volunskuj\interconnect_matrix\interconnect_matrix.ise - [interconnect_matrix.ngc]
File Edit View Project Source Process Window Help
interconnect_matrix
Processes for: interconnect_matrix - Behavioral
Synthesize - XST
View Synthesis Report
View RTL Schematic
View Technology Schematic
Check Syntax
Generate Post-Synthesis Simulati
Implement Design
Generate Programming File
Configure Target Device
Processes
Design Summary
interconnect_matrix.vhd
Synthesis Report
interconnect_matrix.ngr
interconnect_matrix.ngc
Design Objects of interconnect_matrix
Name
Type
data2_out_bus_6_OBUF
Instance
data2_out_bus_7_OBUF
Instance
Properties
No object is selected
Name
Value
Console
Errors
Warnings
Tcl Shell
Find in Files
View by Category
View by Name
[-596,380]

ДОДАТОК И

Лістинг програми розрахунку модулів та їх параметрів для спецпроцесорів різної розрядності

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.IO;
using System.Numerics;
namespace KrestensonFindPrimes
{
   public class PrimesFinder
   {
      private readonly Dictionary<string, uint> _primesList;
      private string _primesCvsPath;

      public bool ShowFullInfo
      {
         get;
         set;
      }
      public bool ShowOnlyCoprime
      {
         get;
         set;
      }
      public PrimesFinder(string primesCvsPath)
      {
         _primesCvsPath = primesCvsPath;

         ShowFullInfo = false;
         ShowOnlyCoprime = false;

         _primesList = _parsePrimesCvsFile(_primesCvsPath);
      }
      private Dictionary<string, uint> _parsePrimesCvsFile(string primesCvsPath)
      {
         var strR = new StreamReader(primesCvsPath);
         var res = new Dictionary<string, uint>();

         do
         {
```

```
            var line = strR.ReadLine();
            if (line == null) break;
            var lineData = line.Split(new[] { ' ' });

            if (Utils.IsBinaryValue(lineData[1]))
            {
               res.Add(lineData[1], uint.Parse(lineData[0]));
            }

         }
         while (true);
         return res;
      }
      public string FindPrimes(int processorCapacity, int blockCapacity, int
additionalValues)
      {
         if (processorCapacity < blockCapacity)
         {
            throw new ArgumentException("processorCapacity повинен бути більшим чи
рівним за blockCapacity");
         }
         if (processorCapacity == 0 || blockCapacity == 0)
         {
            throw new ArgumentException("processorCapacity і blockCapacity повинні
бути більші за 0");
         }
         var primesCount = (int)Math.Round((double)(processorCapacity / blockCapacity),
MidpointRounding.AwayFromZero) + additionalValues;
         if (primesCount % 2 == 1)
         {
            primesCount++;           }
         var res = new StringBuilder();
         var ress = _findPrimes(_primesList, blockCapacity, primesCount);
         if (ShowOnlyCoprime)
         {
            ress = _findCoprimes(ress);
         }
         foreach (var value in ress)
         {
            res.AppendLine(ShowFullInfo ? value.ToString() :
value.GetRealNumber().ToString());           }
         return res.ToString();       }
      private List<ExtendedNumber> _findCoprimes(IEnumerable<ExtendedNumber>
allItems)
      {
```

```
        var existMultipliers = new List<uint>();
        var result = new List<ExtendedNumber>();
        foreach (var number in allItems)
        {
            var multipliers = (from entry in number.GetMultipliers() where entry > 1 select
entry).ToArray();
            var multiplierExist = multipliers.Any(existMultipliers.Contains);
                        if (multiplierExist)
            {
               continue; }
                        foreach (var multiplier in multipliers.Where(multiplier =>
!existMultipliers.Contains(multiplier)))
            {
               existMultipliers.Add(multiplier);              }


            result.Add(number);
        }
        return result;       }
      private List<ExtendedNumber> _findPrimes(Dictionary<string, uint> primesList, int
itemMaxCapacity, int returnPrimesCount)
      {
        var res2 = new List<ExtendedNumber>();
        var res = new List<uint>();
        var searchList = new Dictionary<string, uint>();
        var maxValue1 = (uint)Math.Pow(2, itemMaxCapacity);
        res.Add(maxValue1);
        res2.Add(new ExtendedNumberPowerer(2, (uint)itemMaxCapacity));
        var maxValue = (uint)Math.Pow(2, itemMaxCapacity)-1;
        res.Add(maxValue);
        res2.Add(new ExtendedNumberMultiplicator(Utils.GetMultipliers(maxValue)));
        foreach (var pair in primesList)
        {
          if (pair.Key.Length < itemMaxCapacity)
          {
            searchList.Add(pair.Key, pair.Value);
          }
          else if (pair.Key.Length == itemMaxCapacity)
          {
            if (res.Contains(pair.Value))
              continue;
            res.Add(pair.Value);
            res2.Add(new ExtendedNumberPrime(pair.Value));
          }
          else
```

```
            {
                break;
            }
        }
        var needContinue = res.Count < returnPrimesCount;

        if (needContinue)
        {
            var tempRes = new List<uint>();
            foreach (var pairOuter in searchList)
            {
                uint temp;
                foreach (var pairInner in searchList)
                {
                    if (pairOuter.Key.Length + pairInner.Key.Length - 1 > itemMaxCapacity)
                    {
                        continue;
                    }
                    temp = pairInner.Value * pairOuter.Value;

                    if (temp >= maxValue || tempRes.Contains(temp) || res.Contains(temp))
                        continue;
                    tempRes.Add(temp);
                    res2.Add(new ExtendedNumberMultiplicator(new[] { pairInner.Value,
pairOuter.Value }));
                }
                if (!tempRes.Contains(pairOuter.Value))
                {
                    tempRes.Add(pairOuter.Value);
                    res2.Add(new ExtendedNumberPrime(pairOuter.Value));
                }
                for (var i = 2; i <= itemMaxCapacity; i++)
                {
                    temp = (uint)Math.Pow(pairOuter.Value, i);
                    if (temp > maxValue)
                    {
                        break;
                    }
                    if (tempRes.Contains(temp))
                        continue;
                    tempRes.Add(temp);
                    res2.Add(new ExtendedNumberPowerer(pairOuter.Value, (uint)i));
                }
            }
            res.AddRange(tempRes);
```

```
            }
            res2 = (from entry in res2 orderby entry.GetRealNumber() descending select
entry).ToList();
            return res2.Take(returnPrimesCount).ToList();
        }
    }
}
using System.Globalization;
using System.Linq;
namespace KrestensonFindPrimes
{
    public abstract class ExtendedNumber
    {
        protected uint RealNumber;
        protected uint[] Multipliers;

        public uint GetRealNumber()
        {
            return RealNumber;
        }
        public uint[] GetMultipliers()
        {
            return Multipliers;
        }

    }

    public class ExtendedNumberPrime : ExtendedNumber
    {
        public ExtendedNumberPrime(uint number)
        {
            RealNumber = number;
            Multipliers = new uint[] { number, 1 };
        }
        public override string ToString()
        {
            return RealNumber + " (Просте число)";
        }
    }

    public class ExtendedNumberMultiplicator : ExtendedNumber
    {
        public ExtendedNumberMultiplicator(uint[] multipliers)
        {
```

```
            var number = multipliers.Aggregate<uint, uint>(1, (current, multiplier) =>
current*multiplier);
            RealNumber = number;

            Multipliers = multipliers;
        }

        public override string ToString()
        {
            var res = RealNumber.ToString(CultureInfo.InvariantCulture);
            res += " (Складене число: ";
            res += string.Join(" * ", Multipliers);
            res+=")";

            return res;
        }
    }

    public class ExtendedNumberPowerer : ExtendedNumberMultiplicator
    {
        private readonly uint _powerKoefficient;
        public ExtendedNumberPowerer(uint multiplier, uint powerKoefficient)
            : base(Enumerable.Repeat(multiplier, (int)powerKoefficient).ToArray())
        {
            _powerKoefficient = powerKoefficient;
        }
        public override string ToString()
        {
            return RealNumber + " (Складене число: " + Multipliers[0] +
                " ^ " + _powerKoefficient + ")";
        }
    }
                                                    }
```

Додаток К
Акти впровадження результатів дисертаційної роботи

ТОВАРИСТВО З ОБМЕЖЕНОЮ ВІДПОВІДАЛЬНІСТЮ
«ТЕРНОПІЛЬСЬКЕ КОНСТРУКТОРСЬКЕ БЮРО РАДІОЗВ`ЯЗКУ «СТРІЛА»

46023, м. Тернопіль, вул 15 Квітня, 6, тел/факс – 28-75-00, 28-72-00, tkbr_strila@ukr.net, tkbr_strila@mail.ru
р/р 260070106806 в Тернопільському відділенні №3 ПАТ КБ"Фінансова ініціатива",
МФО 380054, код 14042350
№ 306 від 30.11.2012р.

ЗАТВЕРДЖУЮ

Директор ТОВ ТКБР «Стріла»

Рафалюк О.О.

АКТ

про впровадження результатів дисертаційної роботи аспіранта кафедри спеціалізованих комп'ютерних систем Тернопільського національного економічного університету Волинського Ореста Ігоровича "Методи побудови високопродуктивних спецпроцесорів на основі теоретико-числового базису Крестенсона"

Даний акт складений про те, що результати дисертаційної роботи Волинського О.І, "Методи побудови високопродуктивних спецпроцесорів на основі теоретико-числового базису Крестенсона" передані для впровадження ТОВ ТКБР "Стріла" що включають:

1. Високопродуктивні алгоритми міжбазисних перетворень зеликорозрядних чисел Радемахера-Крестенсона та Крестенсона-Радемахера з використанням розмежованої системи числення.
2. Схемотехнічні рішення спецпроцесорів перетворення великорозрядних шсел двійкової системи числення у систему залишкових класів та їх модульного жспоненціювання, які дозволяють підвищити 1-2 порядки швидкодію цифрування даних у комп'ютерних мережах.

Члени комісії:

Заступник директора Карпів В.Б.

Головний конструктор Піскун С.О.

Затверджую
Проректор з наукової роботи
Тернопільського національного
економічного університету
_____ проф. З.-М.В. Задорожний
«20» «Листопада» 2013р.

АКТ
про впровадження результатів дисертаційної роботи викладача
кафедри спеціалізованих комп'ютерних систем
Волинського Ореста Ігоровича "**Методи побудови високопродуктивних спецпроцесорів на основі теоретико-числового базису Крестенсона"** у науково-дослідній роботі на тему:
"Розробка теорії та комп'ютерних засобів спеціалізованих комп'ютерних систем на основі теоретико-числових базисів Крестенсона-Галуа"
(державний реєстраційний номер 0106U012530)

Ми, комісія у складі завідувача кафедри спеціалізованих комп'ютерних систем Тернопільського національного економічного університету (ТНЕУ) і наукового керівника науково-дослідної роботи д.т.н., професора Николайчука Я.М. та начальника науково-дослідної частини ТНЕУ Письменного В.І., створена для приймання роботи, виконаної в рамках тематичного плану науково-дослідних робіт ТНЕУ на 2012-2013р. на тему "Розробка теорії та комп'ютерних засобів спеціалізованих комп'ютерних систем на основі теоретико-числових базисів Крестенсона-Галуа" (державний реєстраційний номер 0106U012530), встановила:

1. Розроблено Волинським О. І. високопродуктивний однотактний спецпроцесор міжбазисного перетворення Радемахера-Крестенсона шляхом застосування модулів рандомізації та мультиплексування фрагментів розмежованих залишків, що дозволило досягнути максимальної швидкодії спецпроцесора..
2. Розроблено за участю Волинського О. І. структура швидкодіючого спецпроцесора кореляційного опрацювання сигналів у базисі Крестенсона шляхом представлення вхідних аналогових сигналів у базисі Хаара-Крестенсона, що дозволило реалізувати однотактні операції додавання та множення чисел і максимально підвищити швидкодію кореляційного опрацювання сигналів.
3. Розроблено за участю Волинського О. І. процесор шифрування великорозрядних чисел на основі дискретного логарифму для задач криптографії, що дозволило на 1-2 порядки підвищити швидкодію шифрування та штатного дешифрування інформаційних потоків в шифрометодах з відкритими ключами.

Завідувач кафедри спеціалізованих комп'ютерних систем ТНЕУ, д.т.н., проф. Я.М. Николайчук

Науковий керівник науково-дослідної роботи, д.т.н., проф. Я.М. Николайчук

Начальник науково-дослідної частини В.І. Письменний

ЗАТВЕРДЖУЮ
Директор інституту мікропроцесорних систем
керування об'єктами електроенергетики
КД ЦІЗІТ НАН України

_____ к. т. н. Сабадаш І.О.

"03" грудня 2012 р.
м.Львів

АКТ

використання результатів дисертаційної роботи Волинського Ореста Ігоровича " Методи побудови високопродуктивних спецпроцесорів на основі теоретико-числового базису Крестенсона"

Ми комісія у складі:

к. т. н. доцент Кідиба Віктор Павлович (головний інженер), к. т. н. доцент Баран Петро Михайлович (завідувач лабораторії мікропроцесорної техніки), склали даний акт у тому, що за реалізації систем контролю технологічних станів обладнання електричних підстанцій 6-35 кВ використано наступні результати дисертаційної роботи Волинського О.І.:

- методи та алгоритми кореляційного опрацювання сигналів на основі теоретико-числового базису Крестенсона;
- структура та схемотехнічна реалізація високопродуктивного кореляційного спецпроцесора у базисі Хаара-Крестенсона;
- рекурсивний метод міжбазисного перетворення Радемахера-Крестенсона.

Вказані результати використані за розробки алгоритмів та програмно-апаратних засобів встановлення фактів збурень у високовольтних лінія електропередач, що дозволило:

1. Підвищити швидкодію встановлення фактів збурень у високовольтних лінія електропередач.
2. Зменшити апаратні затрати та на 2-3 порядки часову складність кореляційних спецпроцесорів за рахунок виконання матричного множення та сумування цифрових даних у системі залишкових класів.

Ці результати використані за виконання в процесі впровадження інформаційно діагносту вальної системи моніторингу стану ізоляції в мережах 6-35 кВ у підприємстві ВАТ ЕК «Львівобленерго».

Головний інженер
к. т. н. доцент — Кідиба В. П.

Завідувач лабораторії
мікропроцесорної техніки
к. т. н. доцент — Баран П. М.

Затверджую
Проректор з науково-педагогічної роботи
Тернопільського національного
економічного університету
_____ Шинкарик М.І.
«01» грудня 2012р.

АКТ

впровадження результатів дисертаційної роботи Волинського Ореста Ігоровича
«Методи побудови високопродуктивних спецпроцесорів на основі теоретико-числового базису Крестенсона»

Комісія у складі: голови декана факультету комп'ютерних інформаційних технологій, д.т.н., проф. Дивака Миколи Петровича та членів: завідувач кафедри спеціалізованих комп'ютерних систем д.т.н., проф. Николайчука Ярослава Миколайовича, к.т.н., доц. Яцківа Василя Васильовича підтверджує, що результати кандидатської дисертації Волинського Ореста Ігоровича впроваджені і використовуються в навчальному процесі на кафедрі спеціалізованих комп'ютерних систем Тернопільського національного економічного університету при вивчені дисциплін: "Комп'ютерна логіка", "Цифрова обробка сигналів і зображення", "Теорія джерел інформації", "Спецпроцесори в різних теоретико-числових базисах" для студентів спеціальностей 8.05010203 - "Спеціалізовані комп'ютерні системи" та 8.05010201 - "Комп'ютерні системи та мережі", а саме:

- розроблені теоретичні основи виконання арифметичних операцій у розмежованій системі залишкових класів;
- розроблений метод високопродуктивного перетворення чисел з базису Радемахера в базис Крестенсона шляхом рекурентного сканування чисел базису Радемахера;
- вдосконалені цифрові технології перетворення чисел з базису Крестенсона в базис Радемахера на основі цілочисельної та нормалізованої досконалої форм системи залишкових класів шляхом представлення залишків з фіксованою комою у базисі Радемахера;
- вдосконалені цифрові технології перетворення чисел з базису Крестенсона в базис Радемахера на основі цілочисельної та нормалізованої досконалої форм системи залишкових класів шляхом представлення залишків з фіксованою комою у базисі Радемахера
- отримало подальший розвиток кореляційне опрацювання в базисі Хаара-Крестенсона.

Декан факультету комп'ютерних інформаційних технологій ТНЕУ, д.т.н., проф. М.П. Дивак

Завідувач кафедри спеціалізованих комп'ютерних систем ТНЕУ, д.т.н., проф. Я.М. Николайчук

Доцент кафедри спеціалізованих комп'ютерних систем ТНЕУ, к.т.н., доц. В.В. Яцків

Додаток Л
Патенти України на корисну модель

УКРАЇНА

ПАТЕНТ

НА КОРИСНУ МОДЕЛЬ

№ 68872

ПРИСТРІЙ ВИЗНАЧЕННЯ ЗАЛИШКУ
БАГАТОРОЗРЯДНОГО ЧИСЛА

Видано відповідно до Закону України "Про охорону прав на винаходи і корисні моделі".

Зареєстровано в Державному реєстрі патентів України на корисні моделі **10.04.2012.**

Голова Державної служби
інтелектуальної власності України М.В. Паладій

УКРАЇНА

ПАТЕНТ

НА КОРИСНУ МОДЕЛЬ

№ 76623

ПРИСТРІЙ ДЛЯ ПЕРЕТВОРЕННЯ ЧИСЕЛ З ПОЗИЦІЙНОЇ СИСТЕМИ В СИСТЕМУ ЗАЛИШКОВИХ КЛАСІВ

Видано відповідно до Закону України "Про охорону прав на винаходи і корисні моделі".

Зареєстровано в Державному реєстрі патентів України на корисні моделі **10.01.2013.**

Голова Державної служби
інтелектуальної власності України М.В. Ковіня

Державна служба інтелектуальної власності України
Ідентифікаційний код 37552556
№1

УКРАЇНА

ПАТЕНТ

НА КОРИСНУ МОДЕЛЬ

№ 74576

СПОСІБ ВИЗНАЧЕННЯ ЗАЛИШКУ ДВІЙКОВОГО ЧИСЛА

Видано відповідно до Закону України "Про охорону прав на винаходи і корисні моделі".

Зареєстровано в Державному реєстрі патентів України на корисні моделі **12.11.2012.**

Голова Державної служби
інтелектуальної власності України М.В. Ковіня

Державна служба інтелектуальної власності України
Ідентифікаційний код 37552556
№1

УКРАЇНА

ПАТЕНТ

НА КОРИСНУ МОДЕЛЬ

№ 76622

ЦИФРОВИЙ АВТОКОРЕЛЯТОР

Видано відповідно до Закону України "Про охорону прав на винаходи і корисні моделі".

Зареєстровано в Державному реєстрі патентів України на корисні моделі **10.01.2013.**

Голова Державної служби
інтелектуальної власності України М.В. Ковіня

Державна служба інтелектуальної власності України
Ідентифікаційний код 37552556
№1