МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Тернопільський національний економічний університет
Навчально-науковий інститут інноваційних освітніх технологій
Кафедра комп'ютерної інженерії

ОЛЕКСІВ Василь Миколайович

Алгоритм формування перевірочних символів в корегуючих кодах системи залишкових класів / The algorithm for the verification symbols formation verification symbols in the residual classes correction codes system

спеціальність: 123 - Комп’ютерна інженерія
магістерська програма - Комп’ютерна інженерія

Магістерська робота

Виконав студент групи КІм-22
В. М. Олексів

Науковий керівник: д.т.н., професор,
В. В. Яцків

Магістерську роботу допущено до захисту:

ТЕРНОПІЛЬ -2018

РЕЗЮМЕ

Магістерська робота на тему «Алгоритм формування перевірочних символів в корегуючих кодах системи залишкових класів» зі спеціальності 123 «Комп'ютерна інженерія» написана обсягом 81 сторінок і містить 22 ілюстрацій, 6 таблиць, 3 додатки та 55 джерел за переліком посилань.

Метою роботи є підвищення швидкодії формування перевірочних символів в коригуючих кодах системи залишкових класів.

Методи досліджень. Для розв'язання поставлених задач у магістерській роботі використано: теорія інформації та кодування, методи теорії чисел, методи завадостійкого кодування даних, методи і засоби проектування й імплементації цифрових систем на кристалі, мови опису апаратури.

Результати дослідження: розроблено алгоритм обчислення перевірочних символів в коригуючих кодах системи залишкових класів, розроблено структуру кодера.

Результати роботи можуть бути використані при проектуванні завадозахищених комунікаційних систем а також в навчальному процесі.

Орієнтовні напрямки розвитку досліджень: підвищення швидкодії роботи алгоритмів перетворення з системи залишкових класів у позиційну систему числення; оптимізація апаратних затрат на реалізацію кодера.

КЛЮЧОВІ СЛОВА: ФОРМУВАННЯ ПЕРЕВІРОЧНИХ СИМВОЛІВ, КОРИГУЮЧІ КОДИ, СИСТЕМА ЗАЛИШКОВИХ КЛАСІВ, МОДУЛЯРНА АРИФМЕТИКА, ПЕРЕДАЧА ДАНИХ.

RESUME

Master's thesis: "The algorithm for the verification symbols formation in the residual classes correction codes system" from the specialty 123 "Computer engineering" written 81 page volume and contains 22 illustrations, 6 tables, 3 applications and 55 sources for references.

The aim is to increase the speed of the formation of verification symbols in the corrective codes of the system of residual classes.

Research methods. To solve the tasks set in the master's thesis, we use the theory of information and coding, methods of number theory, methods of data-preventing data encoding, methods and means of designing and implementing digital systems on the crystal, and the language of apparatus description.

The Results of the study: An algorithm for calculating verification symbols in the corrective codes of the system of residual classes is developed, the coder structure is developed.

The results of the work can be used in the design of maladjusted communication systems as well as in the learning process.

The approximate directions of research development: increasing the speed of the algorithms of conversion from the system of residual classes to the positional system of calculation; optimization of hardware costs for the implementation of the encoder.

KEYWORDS: FORMATION CHECK SYMBOLS, CORRECTING CODES, RESIDUE NUMBER SYSTEM, MODULAR ARIFMETICS, DATA TRANSMISSION.

ЗМІСТ

ВСТУП

Разом з розвитком високопродуктивних обчислювальних засобів, що застосовуються практично у всіх галузях людської діяльності повільніше розвиваються засоби зв'язку, що забезпечують взаємодію віддалених обчислювальних систем. Широке використання знаходять різні системи зв'язку, починаючи від простої локальної комп'ютерної мережі, закінчуючи складними супутниковими системами обміну інформацією. Вимоги до продуктивності таких систем постійно зростають. Збільшуються об'єми оброблюваної інформації, отже, потрібна висока пропускна спроможність каналів зв'язку, зв'язана не тільки із швидкістю надходження даних, але також з якістю інформації. Протягом всього процесу розвитку засобів зв'язку, забезпеченню перешкодостійкості відводилася особлива увага. Була розроблена велика кількість різних кодів, алгоритмів, пристроїв, призначених для захисту інформації від випадкових спотворень в лініях зв'язку.

Оскільки розвиток кодів, контролюючих помилки, спочатку стимулювався задачами зв'язку, термінологія теорії кодування виникає з теорії зв'язку. Побудовані коди мають також багато інших застосувань. Коди використовуються для захисту даних в пам'яті обчислювальних пристроїв і на цифрових дисках, а також для захисту від неправильного функціонування або шумів в цифрових логічних схемах. Коди використовуються також для стиснення даних, і теорія кодування тісно пов'язана з теорією планування статистичних експериментів.

В багатьох системах зв'язку є обмеження на потужність передавання. Наприклад, в системах ретрансляції через супутники збільшення потужності обходиться дуже дорого. Коди, контролюючі помилки, є чудовим засобом зниження необхідної потужності, оскільки з їх допомогою можна правильно відновити одержані повідомлення. Передача в обчислювальних системах звичайно чутлива навіть до дуже малої частки помилок, оскільки одиночна помилка може порушити програму обчислення. Кодування, що контролює

помилки, стає в цих додатках дуже важливим. Для деяких носіїв обчислювальної пам’яті використання кодів, контролюючих помилки, дозволяє добитися більш щільної упаковки бітів.

Серед всіх існуючих на сьогоднішній день кодів особливу увагу заслуговують згорткові коди [1, 2]. На відміну від інших, вони, володіють невеликою надмірністю, мають високу корегуючу здатність, швидкість декодування і відносну простоту реалізації алгоритмів.

Важливе місце серед кодів здатних виявляти і виправляти помилки займають коди спеціального призначення або коди на основі системи залишкових класів, які потребують подальших досліджень.

Мета і завдання дослідження. Метою роботи є підвищення швидкодії формування перевірочних символів в коригуючих кодах системи залишкових класів.

Досягнення визначеної мети передбачає вирішення таких завдань:

– провести аналіз алгоритмів переводу чисел з системи залишкових класів в позиційну систему числення;

− провести аналіз коригуючих кодів системи залишкових класів;

–розробити алгоритм формування перевірочних символів в коригуючих кодах системи залишкових класів;

− розробити структуру пристрою обчислення перевірочних символів в коригуючих кодах;

− реалізувати на програмованій логічній інтегральній схемі пристрій обчислення перевірочних символів в коригуючих кодах системи залишкових класів;

− провести дослідження пристрою обчислення перевірочних символів.

Об’єкт дослідження – процес формування перевірочних символів в коригуючих кодах системи залишкових класів.

Предмет дослідження – алгоритми формування перевірочних символів в коригуючих кодах системи залишкових класів.

Методи досліджень. Для розв’язання поставлених задач у магістерській роботі використано: теорія інформації та кодування, методи теорії чисел,

методи завадостійкого кодування даних, методи і засоби проектування й імплементації цифрових систем на кристалі, мови опису апаратури.

Наукова новизна одержаних результатів. Удосконалено алгоритм обчислення перевірочних символів в коригуючих кодах системи залишкових класів, що дозволило зменшити апаратні затрати на реалізацію кодера.

Практичне значення отриманих результатів. Розроблено алгоритми, структуру та реалізовано на програмованій інтегральній логічній схемі кодер обчислення перевірочних символів в кодах системи залишкових класів.

Публікації та апробація ДР. Яцків В.В, Олексів В.М. Алгоритм обчислення перевірочних символів в коригуючих кодах системи залишкових класів. Прикладні науково-технічні дослідження: матеріали II міжнар. наук. - практ. конф., 3-5 квіт. 2018 р. – Академія технічних наук України. – Івано-Франківськ : Симфонія форте, 2018. – С.44.

Впровадження результатів МР. Результати роботи прийняті до впровадження в Науково-дослідному інституту інтелектуальних комп’ютерних систем Тернопільського національного економічного університету.

Короткий зміст розділів. В першому розділі проведено аналіз коригуючих кодів системи залишкових класів. Виділено переваги використання коригуючих кодів системи залишкових класів в комунікаційних системах.

Проведено порівняльний аналіз алгоритмів переводу чисел з системи залишкових класів в позиційну систему числення від ефективності яких залежить швидкодія алгоритмів кодування/декодування.

В другому розділі досліджено алгоритми переводу з системи залишкових класів у позиційну систему числення і зворотного перетворення, з метою вибору алгоритму оптимально, за часом перетворення, для обчислення перевірочних символів в коригуючих кодах системи залишкових класів.

Розроблено алгоритм формування перевірочних символів, який використовує поділ двійкового повідомлення на частини і приймає отримані частини, як залишки за вибраними модулями, що дозволяє обробляти вхідні дані подані в позиційній системі числення.

В третьому розділі розроблено кодер на основі запропонованого алгоритму обчислення перевірочних символів. Встановлено, що реалізація кодера на основі запропонованого алгоритму в порівнянні з відомими алгоритмами, забезпечує зменшення апаратних затрат на 20% в залежності від розрядності повідомлення та підвищення швидкодії за рахунок відсутності перетворення повідомлення в систему залишкових класів.

Проведені експерементальні дослідження апаратних затрат (кількість логічних елементів) та часу обчислення перевірочних символів при різній розрядності вхідних даних, для відомих коректуючих кодів системи залишкових класів та для розробленого алгоритму формування коректуючих кодів.

1 АНАЛІЗ КОРЕГУЮЧИХ КОДІВ СИСТЕМИ ЗАЛИШКОВИХ КЛАСІВ

1.1 Основні властивості та переваги системи залишкових класів

Основою для створення обчислювачів, які працюють в системі залишкових класів (СЗК), є можливість одночасно використовувати всі теоретико-цифрові властивості СЗК. Проведемо якісну оцінку їх впливу на методи обробки інформації та структуру обчислювача з урахуванням можливостей корекції помилок, виникаючих в процесі функціонування та передачі даних.

Виділимо три основні властивості СЗК [1, 2].

1. Незалежність залишків. Врахування дані властивості дає можливість побудови спецпроцесора (СП) у вигляді набору з *n* незалежних обчислювальних трактів (ОТ), паралельно функціонуючих в часі. Дана обставина дозволяє розпаралелити алгоритм обчислення на рівні мікрооперацій, що принципово неможливо ні для однієї з існуючих позиційних систем числення (ПСЧ). Це дозволяє реалізувати більшість арифметичних операцій за один такт роботи обчислювача. Очевидно також, що СП в СЗК мають модульну структуру. Це дозволяє здійснювати технічне обслуговування ОТ спецпроцесора не перериваючи вирішення задачі, що особливо важливо для спецпроцесорів, що функціонують в реальному часі [1-4]. Одночасно з цим помилки, що виникли за довільною основою (модулю) m_i СЗК, не «розмножуються", що підвищує достовірність обчислень СП в цілому. При цьому байдуже, чи мали місце одноразові або багаторазові помилки по модулю m_i, або навіть пакет помилок довжиною не більше $[\log_2 (m_i - 1)] + 1$ двійкових розрядів.

Крім цього, помилка, що виникла в будь-якому обчислювальному блоці спецпроцесора в СЗК, зберігається до закінчення обчислень в цьому ж блоці, або самоусувається (наприклад, шляхом множення спотвореного залишку на нуль) в процесі подальшої роботи без зупинки обчислень.

В результаті використання даної властивості СЗК створена унікальна система контролю і корекції помилок в динаміці обчислювального процесу при введенні мінімальної кодової надлишковості [3, 5]. Відмінною рисою такої організації контролю і корекції помилок є можливість їх виявлення та виправлення без зупинки обчислень керуючих алгоритмів, що особливо важливо для систем регулювання і управління об'єктами критичної інфраструктури.

2. Рівноправність залишків. Будь який залишок числа, представленого в СЗК, несе інформацію про його кількісне значення.

Це дає можливість програмними методами замінити не працюючий обчислювальний блок по модулю m_i робочим блоком по модулю m_j ($m_i < m_j$) не перериваючи вирішення задачі. Крім цього, спецпроцесор в СЗК зберігає свою працездатність при відмовах одночасно декількох обчислювальних блоків і здатний виконувати програму при деякому зменшенні точності обчислень, тобто такий спецпроцесор має властивість деградації і є функціонально виключно живучим. В цьому аспекті спецпроцесор в СЗК можна віднести до класу природно відмовостійких обчислювальних структур. Дана властивість обумовлює також наступну характерну особливість функціонування спецпроцесора: один і той же обчислювач в СЗК може мати різну ступінь надійності вирішення завдань в залежності від вимог, що пред'являються до точності, обсягу пам'яті і швидкодії, тобто в процесі вирішення завдань можливо здійснювати "обмінні операції" між точністю обчислень, швидкодією рішення алгоритмів і надійністю функціонування спецпроцесора (достовірністю вирішення завдань) [6, 7-10].

Використання першої і другої властивості обумовлює наявність в спецпроцесорі одночасно трьох видів резервування: структурного, інформаційного та функціонального, що і забезпечує високу відмовостійкість спецпроцесора в СЗК. Результати досліджень показали, що використання властивостей СЗК дозволяє забезпечити не меншу надійність спецпроцесора в класі обчислень, ніж тих які широко застосовуються в позиційних системах числення мажоритарні методи дублювання і потроєння обчислювальних

структур. Це досягається при значно меншому обсязі додатково введеному обладнанні [2, 3, 15 - 20], що підтверджується результатами випробувань існуючих цифрових обчислювальних комплексів і досвідом експлуатації. Аналіз даних властивостей дозволяє також зробити висновок про те, що окремі функціональні блоки і вузли в класі обчислень відносяться до легко контрольованих і діагностованих цифрових пристроїв [11 - 13].

3. Малорозрядність залишків. Створені табличні алгоритми реалізації арифметичних операцій в СЗК дозволяють значно підвищити надійність (за рахунок використання регулярних табличних структур) і продуктивність (це досягається за рахунок можливості розпаралелювання алгоритмів на рівні мікрооперацій) спецпроцесора. Використання табличній арифметики в СЗК дозволило створити оптоелектронні матричні процесори з фіксованою комою і з швидкодією близько сотень мільйонів елементарних операцій в секунду незалежно від величини розрядної сітки обчислювача. В даний час створені і функціонують спецпроцесори в СОК, що виконують роль арифметичних розширювачів (АР) для базових комп'ютерів. В першу чергу АР реалізують арифметичні операції додавання, віднімання і множення, а також вирішують завдання цифрової фільтрації, дискретного перетворення Фур'є і інші.

Проведені дослідження показали, що подання аналогової інформації в СЗК дозволяє створити відмовостійку аналогову обчислювальну систему, що функціонує з будь-якою заданою точністю, тобто забезпечується висока точність обчислень спецпроцесора, яка дорівнює $1/\sum_{i=1}^{n} m_i$, при порівняно невисокій точності обчислень в кожному тракті, що дорівнює $1 / m_i$ [14, 15].

В [16, 17] показано, що голографічні принципи обробки інформації добре узгоджуються з принципами подання та обробки інформації в СЗК. Дана обставина дозволила створити когерентні оптичні обчислювальні структури в СЗК, пристосовані для паралельної обробки інформації, (що обумовлено двовимірної структурою світлового потоку, яка переносить інформацію), векторні і сигнальні процесори, процесори Фур'є, а також зробити висновок про те, що СЗК може бути основою створення систем штучного інтелекту, а також фундаментом при побудові суперкомп'ютерів наступного покоління.

В даний час тривають пошуки шляхів поліпшення окремих характеристик спецпроцесора за рахунок можливості позиційно - залишкового уявлення операндів [5, 6, 7]. Показано, що використання позиційно - залишкового кодування дозволяє знаходити нові оригінальні технічні рішення в області побудови функціональних блоків і вузлів відмовостійких і швидкодіючих спецпроцесорів, що функціонують в реальному часі.

Одним з важливих шляхів подальшого застосування СЗК є використання залишкового кодування при обробці, зберіганні та передачі інформації. Дійсно, відповідно до результатів дослідження корегуючі коди СЗК мають ряд переваг і їх подальше дослідження є актуальною науковою задачею [8 - 9].

1.2 Аналіз алгоритмів відновлення значення числа по його залишках

Модулярна арифметика має ряд переваг в порівнянні з позиційною. Вони полягають в швидкості роботи, паралелізм, вимогам до розрядності чисел. Однак реалізувати всю систему цілком виключно на основі непозиційної арифметики не завжди є доцільним і можливим. Наприклад, коли відбуваються певні розрахунки, то результат в більшості випадків повинен бути представлений позиційно, інакше в ньому може просто не бути сенсу. Крім того, вкрай часто модулярна арифметика використовується спільно з традиційною, доповнюючи й удосконалюючи її. У таких системах швидке виконання переводу чисел в обидві сторони є важливою задачею.

Система залишкових класів (СЗК) є непозиційною системою числення, числа в якій представлені у вигляді наборів залишків від ділення на заздалегідь підібрані модулі (які є взаємно простими між собою числами). Здавалося б у визначенні даної системи чітко прописаний алгоритм переводу в неї: взяти залишки від ділення вихідного позиційного числа на числа, вибрані в якості модулів, і записати їх по порядку. Тим більше більшість мов програмування і проектування обчислювальних систем дану операцію підтримують цілком

непогано. Завдання ускладняється, коли в якості позиційного числа приймаються числа великої розрядності, представлені в так званій "довгій" арифметиці. Дана форма запису числа застосовується при обмеженості розрядної сітки апаратури, використовуваної для обчислень. Якщо розрядна сітка обмежена, припустимо, k бітами, то довге число представляється в системі числення з основою 2k і послідовно записується в пам'ять (наприклад, у вигляді масиву). Завдання отримання залишку від ділення стає в такому випадку нетривіальним. Методи її рішення можна знайти в [10-13].

Система залишкових класів володіє однією особливістю, яку можна віднести до недоліків цієї системи: не можна візуально визначити величину числа, представленого в СЗК, а отже важко і виконання таких операцій, як порівняння чисел, визначення знаку числа. Один із шляхів вирішення цієї проблеми полягає в перетворенні чисел з СЗК в позиційну систему числення. Оцінимо існуючі способи перекладу, як традиційні: метод ортогональних базисів; переклад числа в узагальнену позиційну систему числення (УПСЧ), так і недавно з'явилися інтервальні методи переводу.

1. Метод відновлення числа по його залишках. Даний метод був знайдений ще в Китаї дві тис. років тому. Основою цього методу є теорема, яка має назву Китайська теорема про залишки (КТЗ).

Нехай $p_1,\ p_2,\ ...,\ p_n$ – попарно взаємно прості числа, $P = \prod_{i=1}^{n} p_i$, m_1, $m_2,\ \ldots,\ m_n$ підібрані так, що $\frac{P}{p_i} m_i \equiv 1 (\mathrm{mod}\ p_i)$, $A_0 = \sum_{i=1}^{n} \frac{P}{p_i} m_i \alpha_i,\ i = \overline{1,n}$.

Тоді розв'язок системи $A \equiv \alpha_i (\mathrm{mod}\ p_i),\ i = \overline{1,n}$, буде мати вид: $A \equiv A_0 (\mathrm{mod}\ P)$.

Ця теорема лежить в основі методу ортогональних базисів при переводі з системи залишкових класів в позиційну систему числення.

Нехай модулі системи залишкових класів $p_1,\ p_2,...,\ p_n$; $P = = \prod_{i=1}^{n} p_i$ – діапазон системи. З вибором системи визначаються її основні константи –

базиси B_i, $i = \overline{1,n}$. Задача переводу числа $A = =(\alpha_1, \alpha_2, ..., \alpha_n)$ в ПСЧ полягає в визначенні таких чисел M_i, $i = \overline{1,n}$, щоб $A = \sum_{i=1}^{n} M_i B_i$. Для однозначного визначення на базиси системи накладається ряд обмежень. Базиси володіють властивістю, яка називаються ортогональністю:

$B_1 = (1, 0, 0, \ldots, 0, 0)$, $B_2 = (0, 1, 0, \ldots, 0, 0)$, $B_n = (0, 0, 0, \ldots, 0, 1)$,

При ортогональних базисах $M_i = \alpha_i$, $i = \overline{1,n}$. Ортогональні базиси визначаються за формулою:

$$B_i = \frac{m_i P}{p_i} m_i P_i \ , \ i = \overline{1,n}, \tag{1.1}$$

де

$$P_i = \frac{P}{p_i} = p_1 \cdot p_2 \cdot \ldots \cdot p_{i-1} \cdot p_{i+1} \cdot \ldots \cdot p_n \tag{1.2}$$

m_i – цілі додатні числа, які називаються вагами базису, вони визначаються з рівняння

$$P_i m_i \equiv 1 \pmod{p_i} \tag{1.3}$$

Тоді, за Китайською теоремою, число

$$A = (\alpha_1, \alpha_2, ..., \alpha_n) \equiv \sum_{i=1}^{n} \alpha_i B_i \pmod{P}.$$

Таким чином, якщо знайдені ортогональні базиси для системи модулів, то для переводу числа $A = (\alpha_1, \alpha_2, ..., \alpha_n)$ досить обчислити $\sum_{i=1}^{n} \alpha_i B_i$ і привести цю суму в діапазон [0; P). Відніманням величини, кратної *P*, тобто

$$A = \left| \sum_{i=1}^{n} \alpha_i B_i \right|_p = \sum_{i=1}^{n} \alpha_i B_i - r_A P, \qquad (1.4)$$

де r_A- ранг числа A, який показує скільки разів треба відняти величину діапазону P з отриманого числа, щоб він знаходився в діапазоні.

Розглянемо приклад. Нехай дана система модулів $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, $p_4 = 7$, $p_5 = 11$, значення діапазону $P = 2 \cdot 3 \cdot 5 \cdot 7 \cdot 11 = 2310$. перевести число $A = (1, 2, 1, 4, 7)$ в позиційну систему.

Обчислимо ортогональні базиси.

Для цього знайдемо P_i:

$$P_1 = \frac{P}{p_1} = 1155,\ P_2 = \frac{P}{p_2} = 770,\ P_3 = \frac{P}{p_3} = 462,\ P_4 = \frac{P}{p_4} = 330,$$

$$P_5 = \frac{P}{p_5} = 210.$$

Знайдемо вагу базисів:

$1155\, m_1 \equiv 1 (\mathrm{mod} 2),\ m_1 \equiv 1 (\mathrm{mod}\, 2).$

$770\, m_2 \equiv 1 (\mathrm{mod} 3),\ m_2 \equiv 2 (\mathrm{mod} 3).$

$462\, m_3 \equiv 1 (\mathrm{mod} 5),\ m_3 \equiv 3 (\mathrm{mod}\, 5).$

$330\, m_4 \equiv 1 (\mathrm{mod} 7),\ m_4 \equiv 1 (\mathrm{mod}\, 7).$

$210\, m_5 \equiv 1 (\mathrm{mod} 11),\ m_5 \equiv 1 (\mathrm{mod}\, 11).$

Тепер обчислимо самі базиси:

$B_1 = m_1 \cdot P_1 = 1 \cdot 1155 = 1155,$

$B_2 = m_2 \cdot P_2 = 2 \cdot 770 = 1540,$

$B_3 = m_3 \cdot P_3 = 3 \cdot 462 = 1386,$

$B_4 = m_4 \cdot P_4 = 1 \cdot 330 = 330,$

$$B_5 = m_5 \cdot P_5 = 1 \cdot 210 = 210.$$

Обчислимо значення числа A:

$$A = |1 \cdot 1155 + 2 \cdot 1540 + 1 \cdot 1386 + 4 \cdot 330 + 7 \cdot 210|_{2310} = |8411|_{2310} = 1481.$$

Так як ортогональні базиси повністю визначаються вибором модулів системи, то вони можуть бути заздалегідь обчислені, причому єдиний раз.

Недолік розглянутого методу полягає в тому, що доводиться мати справу з великими числами B_i і, крім того, дії додавання і множення треба виконувати в позиційній системі числення, а з отриманого результату необхідно обчислювати залишок за модулем P.

2). Інший метод визначення величини числа пов'язаний з переведенням числа з системи залишкових класів в узагальнену позиційну систему числення (УПСЧ). Для того щоб розглянути цей метод, виявимо зв'язок між поданням деякого числа A в цих двох системах. Нехай як і раніше СЗК задається модулями $p_1, p_2, ..., p_n$ і $A = (\alpha_1, \alpha_2, ..., \alpha_n)$ - число в цій системі. І нехай $p_1, p_2, ..., p_n$ - також модулі ОПЧ, тоді число A можна представити у вигляді

$$A = a_n p_1 p_2 \dots p_{n-1} + a_{n-1} p_1 p_2 \dots p_{n-2} + \dots + a_3 p_1 p_2 + a_2 p_1 + a_1, \quad (1.5)$$

де $0 \le a_k < \prod_{i=1}^{k-1} p_i \left(i = \overline{1, n}\right)$ – коефіцієнти (цифри) УПЧ.

Очевидно, що діапазони чисел, які представлені в СЗК і УПСЧ збігаються, тобто можна говорити про наявність взаємно однозначної відповідності між множиною представлень чисел в СЗК і УПСЧ. Рівність (1.5) можна записати у вигляді:

$$A = a_1 + p_1(a_2 + p_2(a_3 + \ldots + p_{n-2}(a_{n-1} + p_{n-1}\, a_n)\ldots)),$$

звідки випливає, що цифри УПЧ можуть бути отримані з співвідношень:

$$a_1 = A - \left[\frac{A}{p_1}\right] p_1 = A - A_1\ p_1,$$

де $A_1 = \left[\frac{A}{p_1}\right]$,

$$a_2 = A_1 - \left[\frac{A_1}{p_2}\right] p_2 = A_1 - A_2\ p_2, \text{ где } A_2 = \left[\frac{A_1}{p_2}\right], \qquad (1.6)$$

$$\ldots$$

$$a_n = A_{n-1} - \left[\frac{A_{n-1}}{p_n}\right] p_n = A_{n-1} - A_n\ p_n,$$

де $A_n = \left[\frac{A_{n-1}}{p_n}\right]$.

Причому при визначенні цифр a_i за формулами (1.6) всі обчислення можна проводити в СЗК.

Дійсно, з (1.6) слідує що $a_1 = |A|_{p_1}$, тобто a_1 - перша цифра СЗК або $a_1 = \alpha_1$. Для отримання a_1, спершу $A - a_1$ подамо в залишковому коді. Очевидно $A - a_1$ - ділиться на p_1. Більш того p_1, взаємно просте з усіма іншими модулями. Отже, для знаходження цифри може бути використана процедура розподілу без залишку: $a_2 = \left|\frac{A - a_1}{p_1}\right|_{p_2}$. Таким шляхом, за допомогою обчислень і поділів в залишковому записі всі цифри ОПЧ можуть бути отримані.

При цьому, якщо $a_1 = |A|_{p_1}$, $a_2 = \left| \left[\frac{A}{p_1} \right] \right|_{p_2}$, $a_3 = \left| \left[\frac{A}{p_1 p_2} \right] \right|_{p_3}$

і, взагалі, для $i > 1$ $a_i = \left| \left[\frac{A}{p_1 p_2 \cdots p_{i-1}} \right] \right|_{p_i}$.

Переклад, який здійснюється згідно з алгоритмом (1.6), містить всього $2(n-1)$ залишкові арифметичні операції віднімання і ділення без залишку, де n - число модулів системи. Можна запропонувати деяку модифікацію, тобто операцію ділення замінити операцією множення. Для цього попередньо обчислюється $\frac{n(n-1)}{2}$ констант τ_{kj}, які задовольняють умові:

$$\tau_{kj}\, p_k \equiv 1 (\mathrm{mod}\ p_j),\ 1 \le k < j \le n. \qquad (1.7)$$

Ці константи можна, наприклад отримати з розширеного алгоритму Евкліда:

$$\tau_{kj}\, p_k + \gamma\, p_j = НОД(p_k, p_j) = 1 \qquad (1.8)$$

Тут слід зауважити той факт, що константи τ_{kj} повністю визначаються вибраною системою підстав, тому можуть бути обчислені заздалегідь і зберігатися в деякій таблиці. У додатку до А для модулів p_k і p_j які не перевищують 31 представлена таблиця величин τ_{kj}.

Якщо константи обчислені, то обчислення цифр УПСЧ за алгоритмом (1.6) може бути переписано у вигляді:

$$\begin{aligned}
&a_1 \equiv \alpha_1 (\mathrm{mod}\ p_1),\\
&a_2 \equiv (\alpha_2 - \alpha_1)\ \tau_{12} (\mathrm{mod}\ p_2), \quad (3.9')\\
&a_3 \equiv \big((\alpha_3 - \alpha_1)\ \tau_{13} - \alpha_2\big)\tau_{23} (\mathrm{mod}\ p_3),\\
&\vdots
\end{aligned}$$

$$a_n \equiv \left(\left(...(\alpha_n - \alpha_1)\ \tau_{1n} - \alpha_2\right)\tau_{2n} - ...\alpha_{n-1}\right)\tau_{n-1n} \,(\mathrm{mod}\ p_n).$$

Константи $\tau_{k\,j}$ прийнято також записувати у вигляді $\tau_{k\,j} = \left|\overline{\overline{\frac{1}{p_k}}}\right|_{p_j}$ і називати зворотними елементами по множенню для чисел p_k по модулю p_j (multiplicative inverse).

Розглянемо алгоритм (1.9) на прикладі. Нехай дана система модулів:

$p_1 = 2,\ p_2 = 3,\ p_3 = 5,\ p_4 = 7,\ p_5 = 11$. Значення діапазону $P = 2310$. переведемо число $A = (1, 1, 3, 5, 4)$ в ОПЧ.

Знайдемо спочатку константи $\tau_{k\,j}$:

$$\tau_{12} = \left|\overline{\overline{\frac{1}{2}}}\right|_3 = 2,\ \tau_{13} = \left|\overline{\overline{\frac{1}{2}}}\right|_5 = 3,\ \tau_{14} = \left|\overline{\overline{\frac{1}{2}}}\right|_7 = 4,\ \tau_{15} = \left|\overline{\overline{\frac{1}{2}}}\right|_{11} = 6,$$

$$\tau_{23} = \left|\overline{\overline{\frac{1}{3}}}\right|_5 = 2,\ \tau_{24} = \left|\overline{\overline{\frac{1}{3}}}\right|_7 = 5,\ \tau_{25} = \left|\overline{\overline{\frac{1}{3}}}\right|_{11} = 4,$$

$$\tau_{34} = \left|\overline{\overline{\frac{1}{5}}}\right|_7 = 3,\ \tau_{35} = \left|\overline{\overline{\frac{1}{5}}}\right|_{11} = 9,$$

$$\tau_{45} = \left|\overline{\overline{\frac{1}{7}}}\right|_{11} = 8.$$

Для зручності константи $\tau_{k\,j}$ подамо у вигляді матриці $k \times j$:

$$\begin{pmatrix} 0 & 2 & 3 & 4 & 6 \\ 0 & 0 & 2 & 5 & 4 \\ 0 & 0 & 0 & 3 & 9 \\ 0 & 0 & 0 & 0 & 8 \end{pmatrix}.$$

Виконання алгоритму (1.9) представлено в таблиці 1.1.

Отже,

$$A = a_5 p_1 p_2 p_3 p_4 + a_4 p_1 p_2 p_3 + a_3 p_1 p_2 + a_2 p_1 + a_1 =$$

$$= 7 \cdot 2 \cdot 3 \cdot 5 \cdot 7 + 0 \cdot 2 \cdot 3 \cdot 5 + 1 \cdot 2 \cdot 3 + 2 \cdot 2 + 1 = 1481.$$

Перевага розглянутого методу перед методом ортогональних базисів полягає в тому, що всі обчислення виконуються в модулярній арифметиці, правда, на жаль, не паралельно. Розглянемо деякі модифікації розглянутого методу.

Таблиця 1.1 – Перевід чисел із СЗК в УПЧ

Дія	Модулі					Цифри УПЧ
	$p_1 = 2$	$p_2 = 3$	$p_3 = 5$	$p_4 = 7$	$p_5 = 11$	
A	1	2	1	4	7	$a_1 = \lvert A \rvert_2 = 1$
$- a_1$	$\bar{1}$	1	1	1	1	
$A - a_1$		1	0	3	6	
$\times \tau_{1J}$	0	2	3	4	6	
A_1		2	0	5	3	$a_2 = 2 = \left\lVert \left[\frac{A}{p_1} \right] \right\rVert_{p_2}$
$- a_2$	–	2	2	2	2	
$A_1 - a_2$			3	3	1	
$\times \tau_{2J}$		0	2	5	4	
A_2			1	1	4	$a_3 = 1 =$
$- a_3$		–	$\bar{1}$	1	1	$\left\lVert \left[\frac{A}{p_1 p_2} \right] \right\rVert_{p_3}$
$A_2 - a_3$				0	3	
$\times \tau_{3J}$			0	3	9	
A_3				0	5	$a_4 = 0 =$
$- a_4$				$\bar{0}$	0	$\left\lVert \left[\frac{A}{p_1 p_2 p_3} \right] \right\rVert_{p_4}$
$A_3 - a_4$				0	5	
$\times \tau_{4J}$					8	

Один з варіантів зменшення числа операцій в розглянутому алгоритмі може бути досягнуто шляхом особливого вибору системи модулів. Виберемо систему модулі СЗК наступним чином [14]:

$$p_1,\ p_2 = p_1 + 1,\ p_3 = p_1 \cdot p_2 + 1,\ p_4 = p_1 \cdot p_2 \cdot p_3 + 1,\ldots,\ p_n = \prod_{i=1}^{n-1} p_i + 1.$$

Очевидно, що модулі такої системи взаємно прості числа. Ця система має ту особливість, що $p_j \equiv 1(\mathrm{mod}\, p_k)$ $(1 \le k < j \le n)$, тобто з порівнянь $\tau_{jk}\, p_j \equiv 1(\mathrm{mod}\, p_k)$ отримуємо, що всі константи $\tau_{jk} = 1$. При такому виборі системи модулів алгоритм можна видозмінити таким чином: обчислення починають зі старших модулів, тоді, тому що константи $\tau_{jk} = 1$, то в алгоритмі відпадає необхідність множити на τ_{jk}, тобто:

$$\begin{aligned}
&a_1 \equiv \alpha_1 (\mathrm{mod}\ p'_1),\\
&a_2 = \alpha_2 - a_1 (\mathrm{mod}\ p'_2),\\
&a_3 = \alpha_3 - a_1 - a_2 (\mathrm{mod}\ p'_3),\\
&\quad \ldots\\
&a_n = \alpha_n - \sum_{i=1}^{n-1} a_i (\mathrm{mod}\ p'_n),
\end{aligned} \qquad (1.10)$$

де $p'_1 > p'_2 > p'_3 > \ldots > p'_n$.

Приклад. Виберемо систему модулів за вказаним принципом:

$p_1 = 2,\ p_2 = p_1 + 1 = 2 + 1 = 3,\ p_3 = p_1 p_2 + 1 = 2 \cdot 3 + 1 = 7,\ p_4 = p_1 p_2 p_3 + 1 =$

$= 2 \cdot 3 \cdot 7 + 1 = 43.$

Значення діапазону $P = 2 \cdot 3 \cdot 7 \cdot 43 = 1806.$

Введемо нові позначення:

$p'_1 = p_4 = 43,\ p'_2 = p_3 = 7,\ p'_3 = p_2 = 3,\ p'_4 = p_1 = 2.$

Нехай в системі модулів p'_1, p'_2, p'_3, p'_4 задано число $A = (27,\ 0,\ 1,\ 1)$. Переведемо його в УПСЧ з тією ж системою модулів (таблиця 1.2).

Таблиця 1.2 – Метод переводу чисел з СЗК в УПСЧ на основі вибору системи модулів.

Дія	Модулі				Цифри УПЧ
	$p'_1 = 43$	$p'_2 = 7$	$p'_3 = 3$	$p'_4 = 2$	
A	27	0	1	1	$a_1 = 27$
$-\ a_1$	27	6	0	1	
$A_1 = A - a_1$	–	1	1	0	$a_2 = 1$
$-\ a_2$		1	1	1	
$A_2 = A_1 - a_2$		–	0	1	$a_3 = 0$
$-\ a_3$			0	0	
$A_3 = A_2 - a_3$			–	1	$a_4 = 1$

Отже,

$$(27,0,1,1) = a_4 \cdot p'_1 \cdot p'_2 \cdot p'_3 + a_3 \cdot p'_1 \cdot p'_2 + a_2 \cdot p'_1 + a_1 =$$
$$= 1 \cdot 43 \cdot 7 \cdot 3 + 0 \cdot 43 \cdot 7 + 1 \cdot 43 + 27 = 973.$$

Як видно, при зазначеному виборі системи модулів число операцій зменшується вдвічі, тому що необхідно для здійснення переводу зробити $(n-1)$ операцій віднімання, проти $2(n-1)$ операцій в загальному випадку. Крім того відпадає необхідність обчислювати і зберігати константи. Подібну властивість мають системи модулів:

$$p_1, p_2 = p_1 - 1, p_3 = p_1 \cdot p_2 - 1, ..., p_n = \prod_{i=1}^{n-1} p_i - 1,$$

для яких все константи $\tau_{jk} = -1 \left(1 \le k < j \le n\right)$. Однак, головний недолік зазначених систем полягає в швидкому зростанні модулів системи, так як $p_n = \prod_{i=1}^{n-1} p_i + 1$. Один із способів досягнення компромісу в становищі, що

склалося це вибір такої системи модулів, для якої лише частина констант $\tau_{kj} = \pm 1$.

Інша модифікація алгоритму (1.9) ґрунтується на факті, що якщо в процесі переводу з'являється певна комбінація цифр, то решта цифр числа у УПЧ можна відразу визначити і процес переводу може бути завершений.

1.3 Порівняльний аналіз корегуючих кодів системи залишкових класів

Дослідження, які проводились за останні десятиріччя показали можливість побудови таких систем передавання інформації, в яких за рахунок спеціального кодування може бути створений захист проти самих найрізноманітніших спотворень сигналів, призначених для передачі інформації. Сформувалась точка зору, що боротьба за високу надійність передавання інформації, тобто за достовірність відновлення інформації на приймальному кінці лінії передавання, повинна проводитись не тільки вдосконаленням технічних засобів передачі інформації, де будь-яке можливе підвищення надійності досягається дорогою ціною і потребує розробки складних захисних заходів, але й використанням таких способів кодування інформації, які були б стійкі по відношенню до можливих випадкових спотворень інформації, розуміючи під цим можливість шляхом відповідної обробки прийнятої інформації виключити внесені в неї збурення, очистити її від помилок і досягти повної відповідності тому, що було відправлено в канал зв’язку. Підвищення надійності передачі інформації технічними засобами, навіть якщо не враховувати економічну сторону питання, обмежено рівнем розвитку техніки зв’язку і будь-які досягнення в цій галузі вимагають нових технічних рішень, використання для даної мети спеціальних кодових систем не містить ніяких принципових обмежень. Більше того, при виборі певного коду, який володіє необхідними коригуючими властивостями, можна суттєво знизити вимоги до

надійності самих ліній передачі інформації, зробити їх більш простими і дешевими.

Для обчислювальних засобів використання методів спеціального кодування диктується насущною необхідністю. Так, як будь-яка обчислювальна машина є сама по собі системою передачі і обробки інформації. І обчислювальній машині постійно відбувається циркуляція інформації. Хоча в машині і немає довгих ліній передавання, але по відносно коротких лініях інформація циркулює з великою швидкістю і в великих обсягах. Якщо прийняти певну умовну одиницю, наприклад проходження одним двійковим розрядом одного сантиметра відстані, то впри цих умовних одиницях робота одної обчислювальної машини середньої потужності за фіксований інтервал часу по передачі інформації буде дорівнювати роботі за такий же проміжок часу великим державним телекомунікаційним системам.

Тому навіть з точки зору тільки передачі інформації при розробці обчислювальних засобів виникає важлива задача забезпечення достовірності всього колосального потоку інформації. А в обчислювальній машині, крім цього, повинна бути забезпечена ще і достовірність арифметичної і логічної обробки інформації. Практично без використання методів спеціального кодування забезпечення достовірності в обчислювальній машині досягається подвійним підрахунком для виявлення правильності або неправильності результатів розв’язку задач і потрійним підрахунком в випадку виявленого розходження для вибору правильного результату обчислення. Такий метод забезпечення достовірності зменшує фактичну продуктивність машин в два рази. Звідси зрозуміло, що забезпечення достовірності будь-якими методами, які відрізняються від вказаних, прямо і безпосередньо пов’язано з збільшенням продуктивності обчислювальних машин [15-19].

Для кожного спеціального коду, від якого вимагається, щоб від володів властивістю виявлення і виправлення помилок, характерна наявність двох груп цифр – інформаційної і контрольної. В інформаційну групу входять цифри, які складають числове значення закодованої величини, а в контрольну – цифри, які додатково вводяться для виявлення і виправлення можливих спотворень при

передаванні. Ці додаткові цифри є надлишковими з точки зору числового значення величини і збільшують загальну довжину коду, що в кінцевому варіанті зменшує пропускну здатність каналу при послідовній передачі і збільшує кількість каналів при паралельній передачі. Але ці обставини повинні компенсуватися тими можливостями, які появляються для виявлення і виправлення помилок.

Переважно ця властивість спеціальних позиційних кодів – їх неарифметичність – перешкоджає їх використанню в обчислювальних машинах, оскільки введені контрольні розряди не дозволяють контролювати результати арифметичної операції, в той час як цей контроль для обчислювальної машини не менш важливий, чим контроль передачі інформації.

В даний час в зв'язку з розробкою машинної арифметики в системі залишкових класів виникла можливість побудови непозиційних кодів, які виявляють і виправляють помилки, і разом з тим повністю арифметичних кодів, де інформаційна і контрольна частини повністю рівноправні відносно будь-якої операції.

Розглянемо систему з основами $p_1, p_2, \ldots, p_n$ і діапазоном $\wp = p_1 \cdot p_2 \cdot \ldots \cdot p_n$. В подальшому діапазон $\wp$ будемо називати робочим діапазоном. Введемо основу p_{n+1} взаємно просту з будь-якою із прийнятих основ і будемо представляти числа в системі із основами $n+1$. Це означає, що будемо передавати числа і виконувати операції над числами, які знаходяться в діапазоні $[0, \wp]$ в більш широкому діапазоні $[0, P]$, де $P = \wp \cdot p_{n+1}$.

В подальшому діапазон P будемо називати повним діапазоном системи з однією контрольною основою.

Так як всі числа з якими працює обчислювальна машина, повинні знаходитись в діапазоні $[0, \wp]$, то зрозуміло, якщо в результаті будь-якої операції або при передаванні числа вийшло, що одержано число А, більше $\wp$, це означає, що при виконанні операції була допущена помилка.

Отже, числа менші $\wp$ будемо називати правильними, а більші $\wp$ – неправильні.

Нехай основи $p_1, p_2, \ldots, p_n, p_{n+1}$ системи залишкових класів задовольняють умові

$$p_i < p_{n+1},$$

$$i = 1, 2, \ldots, n,$$

і нехай $A = (\alpha_1, \alpha_2, \ldots, \alpha_i, \ldots, \alpha_n, \alpha_{n+1})$ – правильне число.

Тоді число $\tilde{A} = \left(\alpha_1, \alpha_2, \ldots, \tilde{\alpha}_i \neq \alpha_i, \ldots, \alpha_n, \alpha_{n+1} \right)$, де $i = 1, 2, \ldots, n, n+1$ є неправильне число.

Отже, будь-яке спотворення цифри по одному розряді перетворює це число в неправильне і відповідно дозволяє виявити наявність помилки. Більше того, існує тільки одно значення цієї цифри, яке може перетворити неправильне число в правильне.

Контрольна основа p_{n+1} повинна бути більшою за будь-яку іншу основу системи.

Розглянемо приклад. Виберемо систему з основами $p_1 = 2$, $p_2 = 3$, $p_3 = 5$, $p_4 = 7$, для яких діапазон правильних чисел (робочий діапазон) $\wp = 2 \cdot 3 \cdot 5 \cdot 7 = 210$. Введемо контрольну основу $p_5 = 11$. Тоді повний діапазон визначається як: $p = 210 \cdot 11 = 2310$.

Обчислимо ортогональні базиси системи:

$$B_1 = (1, 0, 0, 0, 0) = 1155;\ B_2 = (0, 1, 0, 0, 0) = 1540;\ B_3 = (0, 0, 1, 0, 0) = 1386;$$
$$B_4 = (0, 0, 0, 1, 0) = 330;\ B_5 = (0, 0, 0, 0, 1) = 210.$$

Приклад. Передано число $A = (1, 2, 2, 3, 6) = 17$. Замість нього прийнято число $\tilde{A} = (1, 2, 2, 5, 6)$. Для виявлення помилки обчислюємо величину $\tilde{A}$:

$$\tilde{A} = 1 \cdot 1155 + 2 \cdot 1540 + 2 \cdot 1386 + 5 \cdot 330 + 6 \cdot 210 - r2310 = .$$
$$= 9917 - 9240 = 677 > 210$$

Оскільки прийняте число $\tilde{A}$ більше 210, воно є неправильним. Що і вказує на наявність помилки при передаванні числа.

Приклад. Передано число 17. Прийнято $\tilde{A} = (1, 2, 2, 3, 0)$ спотворена цифра контрольного розряду. Обчислюємо величину $\tilde{A}$:

$$\tilde{A} = 1 \cdot 1155 + 2 \cdot 1540 + 2 \cdot 1386 + 3 \cdot 330 + 0 \cdot 210 - r2310 =$$
$$= 7997 - 6930 = 1067 > 210.$$

Отже, $\tilde{A} > 210$, що вказує на наявність помилки.

Приклад. Замість числа А=17, прийнято число $\tilde{A} = (0, 2, 3, 3, 6)$. В цьому прикладі спотворення відбулись по двох основах 2 і 5. Так як $\bar{p} = 2 \cdot 5 = 10 < p_5$, то це спотворення повинно бути виявлено.

Дійсно:

$$\tilde{A} = 0 \cdot 1155 + 2 \cdot 1540 + 3 \cdot 1386 + 3 \cdot 330 + 6 \cdot 210 - r2310 =$$
$$= 9488 - 9240 = 248 > 210.$$

Отже, число $\tilde{A}$ – неправильне число.

Розглянемо приклад, коли по основах, добуток яких більший 11, помилки виявити неможливо.

Приклад. Замість числа А=17 прийнято $\tilde{A} = (1, 2, 2, 0, 0)$.

$$\tilde{A} = 1 \cdot 1155 + 2 \cdot 1540 + 2 \cdot 1386 + 0 \cdot 330 + 0 \cdot 210 - r2310 =$$
$$= 7007 - 6930 = 77 < 210.$$

Отже, $\tilde{A}$ одержали правильним числом. Спотворення в двох розрядах не було виявлено.

Дослідження завадостійких кодів проведемо на основі порівняння основних характеристик кодів до яких відносяться:

- число дозволених і заборонених кодових комбінацій;
- надлишковість коду;
- мінімальна кодова відстань;
- число помилок, що виявляються або виправляються;
- коригуючі можливості кодів.

Проведемо дослідження кодів Хемінга (12, 8), БЧХ (15,11) і СЗК.

1. Число дозволених і заборонених кодових комбінацій.

Для блокових двійкових кодів, з числом символів в блоках рівним n, загальне число можливих кодових комбінацій визначається значенням

$$N_0 = 2^n. \tag{1.10}$$

Число дозволених кодових комбінацій за наявності k інформаційних розрядів в первинному коді рівне

$$N_k = 2^k. \tag{1.11}$$

Очевидно, що число заборонених комбінацій рівне:

$$N_3 = N_0 - N_k = 2^n - 2^k \tag{1.12}$$

а з врахуванням $r = n - k$ відношення буде:

$$\frac{N_0}{N_k} = \frac{2^n}{2^k} = 2^n - 2^k = 2^r \tag{1.13}$$

де r - число надлишкових (перевірочних) розрядів в блоковому коді.

Розрахуємо відношення загального числа кодових комбінацій до числа інформаційних комбінацій:

– код Хемінга – 16;

– код БЧХ – 16;

– код СЗК – 11.

Число заборонених комбінацій рівне:

– код Хемінга – 3840;

– код БЧХ – 30720;

– код СЗК – 2100.

2. Надлишковість завадостійкого коду.

Надлишковістю завадостійкого коду називають величину

$$\chi = \frac{r}{n} = \frac{n-k}{n} = 1 - \frac{k}{n} \qquad (1.14)$$

звідки слідує

$$B_k = \frac{k}{n} = 1 - \chi . \qquad (1.15)$$

Ця величина показує, яку частину загального числа символів кодової комбінації складають інформаційні символи. В теорії кодування величину B_k називають відносною швидкістю коду.

Розрахуємо надлишковість вказаних кодів (1.14):

– код Хемінга – 0,333333;

– код БЧХ – 0,266667;

– код СЗК – 0,309605.

Відповідно, відносна швидкість кодів (1.15) дорівнює:

– код Хемінга – $B_k = 0.733333$;

– код БЧХ – $B_k = 0.666667$;

– код СЗК – $B_k = 0.690395$.

Розрахуємо кодові відстані d_{min} при заданому числі розрядів n в кодовій комбінації і числі інформаційних розрядів k, для кодів:

– Хемінга $d_{min} \le 6.0235$;

– БЧХ $d_{min} \le 7.50366$;

– СЗК $d_{min} \le 5.6135$.

Існуючі методи побудови завадостійких кодів вирішують в основному задачу знаходження такого алгоритму кодування і декодування, який дозволяв би найбільш просто побудувати і реалізувати код із заданим значенням d_{min}. Тому різні коректуючі коди при однакових d_{min} порівнюються по складності кодуючого і декодуючого пристроїв. Цей критерій є у ряді випадків визначаючим при виборі того або іншого коду.

Важливою перевагою кодів системи залишкових класів є їх арифметичність, тобто здатність контролювати правильність виконання будь-яких математичних операцій.

2 АЛГОРИТМ ФОРМУВАННЯ ПЕРЕВІРОЧНИХ СИМВОЛІВ В КОРЕГУЮЧИХ КОДАХ

2.1 Перевід чисел з системи залишкових класів в позиційну систему числення

Перевід чисел з системи остаточних класів у двійкову позиційну систему виконується послідовним відніманням констант, що представляють в системі залишкових класів степені двійки. Двійкові цифри результату формуються по черзі з боку старшого розряду. Знак різниці числа і відповідної степені двійки визначається за допомогою методу нулевізації.

Метод нулевізації полягає в послідовному складанні вихідного числа

$$A = (a_1, a_2, \ldots, a_n),$$

з деякими мінімальними числами, внаслідок чого вихідне число перетвориться в числа вигляду [20-24]:

$$\left(a_1', 0, a_3', \ldots, a_n'\right),\ \left(a_1'', 0, 0, a_3'', \ldots, a_n''\right),\ \ldots,\ \left(e_1, 0, 0, \ldots, 0\right).$$

Цифра e1 по першій підстановці показує, якому з інтервалів [0, P) або [P, $\wp$) належить число A і, отже, вказує на знак числа.

Кількість чисел нулевізації:

$$p_2 + p_3 + \ldots + p_n \cdot \bar{n} \cdot (n-1),$$

кількість кроків нулевізації (n – 1).

Константи нулевізації залежать лише від підстав СОК і прораховуються заздалегідь.

Розглянемо алгоритм перетворення числа, заданого в СЗК, в позиційну систему числення. Нехай задані модулі $(p_1, p_2, ..., p_{k-1}, p_k)$. Задамо k чисел $B_1, B_2, ..., B_{k-1}, B_k)$ в СЗК таких, що $B_j = (\beta_1^j, \beta_2^j, ..., \beta_{k-1}^j, \beta_k^j)$, $j = \overline{1,k}$, які будемо називати базисами даної системи. Їх значення визначаються однозначно з вибором набору модулів і відомі як в СЗК так і в позиційній системі числення. Припустимо, що для переведення з СЗК, задано число $A = (\alpha_1, \alpha_2, ..., \alpha_{k-1}, \alpha_k)$.

Задача переведення полягає в тому, щоб знайти числа μ_i, $i = \overline{1,k}$ такі, що:

$$\sum_{i=1}^{k} \mu_i \cdot B_i = A. \tag{2.1}$$

В системі залишкових класів (2.1) набуде вигляду

$$\sum_{i=1}^{k} \mu_i \cdot (\beta_1^i, \beta_2^i, ..., \beta_{k-1}^i, \beta_k^i) = (\alpha_1, \alpha_2, ..., \alpha_{k-1}, \alpha_k).$$

Прирівнявши залишки по відповідних модулях отримаємо систему лінійних алгебраїчних рівнянь:

$$\begin{cases} \mu_1 \beta_1^1 + \mu_2 \beta_1^2 + ... + \mu_k \beta_1^k = \alpha_1 \\ \mu_1 \beta_2^1 + \mu_2 \beta_2^2 + ... + \mu_k \beta_2^k = \alpha_2 \\ .... \quad ... \quad ... \\ \mu_1 \beta_k^1 + \mu_2 \beta_k^2 + ... + \mu_k \beta_k^k = \alpha_k \end{cases}. \tag{2.2}$$

Система має цілий розв'язок, якщо:

$$\begin{vmatrix} \beta_1^1 & \beta_1^2 & ... & \beta_1^k \\ \beta_2^1 & \beta_2^2 & ... & \beta_2^k \\ ... & ... & ... & ... \\ \beta_k^1 & \beta_k^2 & ... & \beta_k^k \end{vmatrix} = \pm 1. \tag{2.3}$$

Очевидно, що існує безліч розв'язків $(\beta_1^i, \beta_2^i, ..., \beta_k^i)$, $i = \overline{1,k}$. Виберемо частковий розв'язок:

$$\begin{cases} B_1 = (1,0,...,0) \\ B_2 = (0,1,...,0) \\ ... \quad ... \quad ... \\ B_k = (0,0,...,1) \end{cases}, \tag{2.4}$$

який є одним з ортогональних базисів системи. Відповідно до системи рівнянь (2.2) для ортогональних базисів виконується умова

$$\mu_i = \alpha_i, \quad i = \overline{1,k}. \tag{2.5}$$

Звідки

$$A = \sum_{i=1}^{k} \alpha_i \cdot \beta_i \pmod{\wp}. \tag{2.6}$$

Згідно визначення ортогональних базисів (1.10), вони можуть бути обчислені:

$$B_i = m_i \cdot \frac{\wp}{p_i}, \; i = \overline{1,k}; \tag{2.7}$$

де $1 \le m_i \le p_i - 1$ – вага ортогонального елементу.

При чому

$$m_i \cdot \frac{\wp}{p_i} = 1 \pmod{p_i}. \tag{2.8}$$

Рівняння (2.8) еквівалентне наступному діафантовому рівнянню:

$$m_i \cdot \frac{\wp}{p_i} = 1_i \cdot p_i + 1, \quad 1_i \in N . \tag{2.9}$$

Для обчислення m_i використовується формула (2.8). Застосування операції визначення залишку по заданому модулю обумовлює обмежений діапазон можливих значень вагових коефіцієнтів: $m_i \in [1, p-1]$.

Позначимо $\wp_i = \frac{\wp}{p_i}$. В результаті ділення $\wp_i$ на p_i отримаємо певний залишок δ_i, згідно рівняння (2.8):

$$m_i \cdot \delta_i = 1 \cdot (\mathrm{mod}\wp). \tag{2.10}$$

З огляду на порівняно невеликі значення величини p_i можемо скласти таблицю розв'язків рівняння (2.10), за допомогою якої згідно величини δ_i знаходиться відповідне значення m_i. Припускаючи, що основи p_i вибираються з множини простих чисел, приведемо таблицю розв'язків рівняння (2.10), для $p_i < 25$ (таблиця 2.2).

Згідно (2.4):

$$B_1 + B_2 + ... + B_k = (1,0,...,0) + (0,1,...,0) + ... + (0,0,...,1) = (1,1,...,1) . \tag{2.11}$$

Оскільки сумування проводиться в СЗК:

$$\sum_{i=1}^{k} B_i = 1 \pmod{\ldots \wp} . \tag{2.12}$$

Рівняння (2.12) можна використати для перевірки достовірності знаходження базисів системи.

Однією з основних переваг цілочисельного перетворення СЗК є незалежність розрядів числа, що створює можливості для розпаралелення

обробки інформації і підвищення загальної продуктивності обчислювальних засобів.

Таблиця 2.2 – Розв'язки рівняння $m \cdot \delta = 1(\mathrm{mod} p)$ для множини простих чисел $p_i < 25$

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

Особливістю даних в СЗК є малорозрядність представлення числа, що призводить до зменшення розрядності самих обчислювальних засобів.

Недоліком СЗК є обмеженість діапазону чисел, що може бути представлений за допомогою набору $((p_1, p_2,..., p_{k-1}, p_k)$. Даного недоліку легко уникнути, задаючи на початковому етапі модулі, які забезпечують необхідний діапазон.

2.2 Перевід чисел з позиційної системи числення в систему залишкових класів

На даний час існує багато алгоритмів переведення чисел з позиційної системи числення в систему залишкових класів. Дані методи різняться за швидкістю виконання операції переведення і за складністю, що безповоротно впливає на величину апаратних затрат на їх реалізацію.

В даному розділі будуть приведені теоретичні основи різних методів переведення даних з позиційних систем числення в систему залишкових класів.

Для реалізації обчислювального процесу з використанням поліноміальної системи класів залишків необхідно здійснити перетворення з позиційної системи в модулярний коди і назад [25- 29]. Такі операції є немодульними і відносяться до класу позиційних операцій, які є найбільш трудомісткими в непозиційній системі залишкових класів. Як правило, немодульні процедури реалізують за допомогою послідовності модульних операцій.

Однією з перших немодульних процедур, необхідних для функціонування спецпроцесора залишкових класів, є реалізація прямого перетворення позиційних кодів в код поліноміальної системи залишкових класів розширеного поля Галуа GF(p^v) [30 - 34].

Всі існуючі методи переведення з позиційної системи числення в систему залишкових класів можна звести до трьох основних груп.

У основу методів першої групи покладений метод пониження розрядності числа, що не містить операцію ділення.

У основу даного методу покладена теорема, згідно якої обчислення залишку здійснюється за допомогою ітераційного алгоритму. Для цього необхідно визначити залишки від ділення на p_j степенів основи, які дадуть набір чисел C_i, i=1, 2..., r. Якщо залишок від ділення степені основи C_i перевершує половину модуля p_j, то як значення C_i необхідно узяти число, яке доповнює до значення p_j, із знаком мінус. Значення C_i можна знати заздалегідь і вони є константами для вибраної системи числення.

Кількість розрядів C_i визначається розрядністю вихідного числа А. Потім цифри вихідного числа умножаються на відповідні числа C_i, отримана сума визначається

$$A_1 = A_k \cdot C_k + \ldots + A_1 \cdot C_1 + A_0 \cdot C_0 < A_k \cdot S^k + \ldots + A_1 \cdot S^1 + A_0. \qquad (2.1)$$

У цьому методі використовується два принципи. Перший полягає в перетворенні числа А більшій розрядності в число малої розрядності за рахунок використання в якості коефіцієнтів

$$a_i = C_1 = \left|2^i\right|_{p_i}^{+} \le p,$$

розрядність яких не перевищує розрядності модуля p_i. Друга ідея полягає в знаходженні згортки вихідного числа шляхом визначення найменшого невід'ємного залишку в результаті реалізації першої ідеї малорозрядного числа послідовним використанням розробленого методу до здобуття операції скорочення по модулю p_i, тобто

$$\left|A\right|_{p_i}^{+} \le p_i .$$

Основу другої групи складають методи, що забезпечують просторовий розподіл обчислювального процесу – переводу з ПСЧ в ПЧЗК. Число шарів в такій мережі визначається кількістю ітерацій l, необхідних для перетворення вхідних даних, а кількість нейронів в кожному шарі – розрядністю оброблюваних даних на кожній з ітерацій [34].

В цьому випадку ітеративний алгоритм перетворення А по модулю р визначається виразом

$$A(l+1) = \sum_{i=0}^{ordA(l)} \left|2^i\right|_p^{+} \cdot \left\lfloor \frac{A(l)}{2^i} \right\rfloor_2^{+}, \qquad (2.2)$$

де $l = 0,1,2,\ldots$ - число ітерацій.

Заміна зворотних зв'язків в нейронних мережах на прямі дозволяє підвищити швидкість обробки даних, оскільки в такій мережі одночасно обробляється декілька відліків і в кожному такті роботи мережі на вході формуються перетворені дані. Максимальне значення числа на першій ітерації max {A(l)} можна визначити в припущенні, що число складається з одних одиниць [35- 39].

Обчислювальні процеси третьої групи методів переду чисел з ПСЧ в непозиційну систему реалізують різні варіанти методу безпосереднього додавання [40 - 44].

Перетворення вихідного A(z), заданого в розширеному полі GF(p^v), в поліноміальну систему залишкових класів здійснюється за допомогою набору констант, що є еквівалентами степенів основ 2^i і коефіцієнтів, при відповідних степенях основ a^i, представлених в системі залишкових класів.

Перевід з позиційної двійкової коди в поліноміальну систему залишкових класів здійснюється за формулою

$$a(z) \equiv A(z) \bmod p_i(z) = l = 0 \sum^{k} a_i(z) \cdot z^l \cdot \bmod p_i(z),$$

де $i = 1, 2, \ldots, n$.

Для отримання $A(z)$ в системі залишкових класів з основами p1(z), p2(z)..., pn(z) необхідно обчислити в цій системі значення

$$a_l(z) \cdot z^l \cdot \bmod p_i(z).$$

В цьому випадку залишок по модулю $p_i(z)$ визначається

$$a_i(z) = \left| \sum_{l=0}^{k} \left(a_l^i \cdot z^l \right) \bmod p_i(z) \right|_2^+ \qquad (2.4)$$

де $a_l^i = a_l \cdot \bmod p_i(z)$, $i = 1, 2, \ldots, n$.

Відповідно до виразу (2.4), переведення A(z) з позиційної системи числення в непозиційну можна звести до додавання по модулю два величин $\left(a_l^i \cdot z^l \right) \cdot \bmod p_i(z)$ відповідно до заданого полінома A(z) [5 - 9].

Таким чином модифікація і реалізація методу безпосереднього додавання для поліноміальної системи залишкових класів дозволяє розробляти високошвидкісні перетворювачі кодів для обчислювальних структур реального масштабу часу.

При невеликому діапазоні представлених даних найбільш ефективним є табличний метод кодування та перетворення даних в СЗК.

Нехай у десятковій системі числення задано число N=103, вибираємо взаємно прості модулі: $p_1 = 3, \quad p_2 = 5, \quad p_3 = 7$, добуток яких

$$\wp = \prod_{i=1}^{3} p_i = 3 \cdot 5 \cdot 7 = 105.$$

Враховуючи, що $N < \wp$ можна використовувати даний набір модулів для перетворення (таблиця 2.1).

Табличний метод. Отже, згідно таблиці 2.1: $101_{10} = (2, 1, 3)_{(3,5,7)}$.

Таблиця 2.1 – Таблиця кодування даних в СЗК

Число в десятковій системі числення	$p_1 = 3$	$p_2 = 5$	$p_3 = 7$
0	0	0	0
1	1	1	1
2	2	2	2
3	0	3	3
4	1	4	4
5	2	0	5
6	0	1	6
7	1	2	0
8	2	3	1
9	0	4	2
10	1	0	3
11	2	1	4
12	0	2	5
13	1	3	6
14	2	4	0
15	0	0	1
…	…	….	….
103	1	3	5
104	2	4	6
105	0	0	0

Розрахунковий метод. Використовуючи рівняння (2.1) маємо:

$$b_1 = res\,101\ (\mathrm{mod}\,3) = 2;$$

$$b_2 = res\,101\ (\mathrm{mod}\,5) = 1;$$

$$b_2 = res\,101\ (\mathrm{mod}\,7) = 3.$$

Отже $101_{10} = (2,1,3)_{(3,5,7)}$.

Число N представлено в позиційній системі числення з основою $d = 10$. Представлення степенів основи d в СЗК буде мати вигляд:

$$d^0 = 1 = (1,1,1)_{(3,5,7)},$$

$$d^1 = 10 = (1,0,3)_{(3,5,7)},$$

$$d^2 = 100 = (1,0,2)_{(3,5,7)}$$

Отримаємо представлення коефіцієнтів полінома (2.2).

$$a_0 = 1 = (1,1,1)_{(3,5,7)},$$

$$a_1 = 0 = (0,0,0)_{(3,5,7)},$$

$$a_2 = 1 = (1,1,1)_{(3,5,7)}.$$

Згідно формули (2.2):

$$101_{10} = (1\cdot 1 + 1\cdot 0 + 1\cdot 1, 1\cdot 1 + 0\cdot 0 + 1\cdot 0, 1\cdot 1 + 0\cdot 3 + 1\cdot 2) = (2,1,3)_{(3,5,7)}.$$

Представлення числа $N=101_{10}$ отримані за допомогою методів 1, 2 та 3 аналогічні, що підтверджує достовірність отриманих результатів.

Розглянемо приклад зворотного перетворення для значень отриманих вище

$$P_1 = 3, P_2 = 5, P_3 = 7.$$

$$a_1 = 1, a_2 = 3, a_3 = 5.$$

$$\delta_1 = 35(\text{mod}3) = 2,$$

$$\delta_2 = 21(\text{mod}5) = 1,$$

$$\delta_3 = 15(\text{mod}7) = 1,$$

Використовуючи означення базисних чисел та таблицю 2.1:

$$m_1 = 2;\ m_2 = 1;\ m_3 = 1;$$

$$B_1 = \frac{105}{3} \cdot 2 = 70;$$

$$B_2 = \frac{105}{5} \cdot 1 = 21;$$

$$B_3 = \frac{105}{7} \cdot 1 = 15.$$

Перевіримо достовірність обчислення базисних чисел згідно формули (2.3):

$$(70 + 21 + 15) = 106 = 1 \pmod{105}.$$

Згідно формули (2.4):

$$N_{10} = res(2 \cdot 70 + 1 \cdot 21 + 3 \cdot 15) \pmod{105} = 101_{10}.$$

В результаті послідовного застосування прямого та зворотного перетворень для цілочисельної форми СЗК отримаємо вихідне число в позиційній системі числення.

2.3 Алгоритм формування перевірочних символів

В роботі пропонується новий спосіб формування перевірочних символів в коригуючих кодах СЗК, суть якого полягає в наступному. Послідовність бітів, яка підлягає передачі, розділяється на k частин по 4 або 8 біт:

$$\left(a_1^1 \dots a_i^1 \dots a_m^1,\ a_1^2 ,\dots a_i^2 \dots a_m^2, \cdots, a_1^j\ a_2^j \dots a_i^j \dots a_m^j, \cdots, a_1^k \dots a_i^k \dots a_m^k\right), \quad (2.5)$$

де a^i – розряд даних в двійковому коді, $m = 4, 8$.

Кожній частині двійкового коду ставляться у відповідність прості числа (модулі) p_i ($p_1 < p_2 < \dots < p_i < \dots < p_n$) з яких перші k модулів інформаційні, n – загальна кількість модулів, $r = n - k$ – перевірочні модулі. Значення модулів вибираємо з умови $p_i > 2^m$. При цьому перші k модулів визначають робочий діапазон $P_k = \prod_{i=1}^{k} p_i$, повний діапазон дорівнює $P = \prod_{i=1}^{n} p_i$.

Так як значення тетрад або байтів в позиційному представленні менші, за відповідні модулі p_i, то їх можна вважати залишками. В результаті вказаного перетворення повідомлення набуде вигляду:

$$(x_1, x_2, \dots\ x_i, \dots, x_k), \quad (2.6)$$

де x_i – частини повідомлення, які одночасно є залишками по вибраних модулях p_i, $x_i = \sum_{i=1}^{m} a_i \cdot 2^i$.

Для обчислення перевірочних символів повідомлення (2.6) перетворимо в позиційну систему числення

$$X = \sum_{i=1}^{k} (x_i \cdot M_i \cdot \delta_i) \bmod P_k, \tag{2.7}$$

де $M_i = \frac{P_k}{p_i}$, $\delta_i = M_i^{-1} \bmod p_i$.

Перевірочні символи обчислюються за формулою [45]:

$$x_{k+i} = X \bmod p_{k+i},\ i = \overline{1, (n-k)},$$

де X – повідомлення в позиційній системі числення,

В результаті кодове слово складається з інформаційних і перевірочних символів і має вигляд:

$$(x_1, x_2, \ldots x_i, \ldots x_k, x_{k+1}, \ldots x_n).$$

Для використання відомих корегувальних кодів СЗК необхідно перевести вхідне повідомлення в систему залишкових класів за формулою [21, 46]:

$$x_i = X \bmod p_i,\ i = \overline{1, k},$$

а після виявлення та виправлення помилок виконати обернене перетворення за формулою (2.3), що потребує додаткових затрат часу.

Приклад. Нехай $X = 1010011101011001$ – повідомлення, яке необхідно передати. Розділимо дане повідомлення X на чотири тетради: $x_1 = 1010$, $x_2 = 0111$, $x_2 = 0101$, $x_4 = 1001$. Виберемо модулі, згідно умови $p_i > 2^4$: $p_1 = 17$, $p_2 = 19$, $p_3 = 23$, $p_4 = 29$ – інформаційні, $p_5 = 31$ – перевірочний модуль. Робочий діапазон становить $P_k = 17 \cdot 19 \cdot 23 \cdot 29 = 215441$. Загальний діапазон $P = P_k \cdot p_5 = 215441 \cdot 31 = 6678671$.

Оскільки значення x_1, x_2, x_3, x_4 в десятковій системі числення менші за відповідні модулі, то їх будемо вважати залишками за даними модулями.

Переведемо повідомлення $X = (x_1, x_2, x_3, x_4)$ в десяткову систему числення. Для цього обчислимо ортогональні базиси: $M_1 = \frac{P_K}{p_1} = 12673$, $M_2 = 11339$, $M_3 = 9367$, $M_4 = 7429$. Обернені числа до $M_1 \div M_4$ рівні $\delta_1 = 15$, $\delta_2 = 14$, $\delta_1 = 4$, $\delta_4 = 6$. Отже, $X = \sum_{i=1}^{k} (x_i \cdot M_i \cdot \delta_i) \bmod P_K = 153622$.

Перевірочний символ обчислюється за формулою

$$x_5 = X \bmod p_5 = 153622 \bmod 31 = 17.$$

Таким чином, повідомлення після кодування має вигляд:

$$X' = (10,\ 7,\ 5,\ 9, 17),$$

або

$$X' = (1010,\ 0111,\ 0101,\ 1001, 10001).$$

Перевагою розробленого алгоритму формування перевірочних символів є те, що вхідні дані залишаються подані в позиційній системі числення.

Запропонований алгоритм формування перевірочних символів в коригуючих кодах системи залишкових класів приведений на рисунку 2.1.

В блоці 1 здійснюється ввід даних, зоктема самого повідомлення та інформаційних і перевірочних модулів. В блоці 2 відбувається розділення повідомленя на частини, як правило на 4, 8 або кратне 4 число біт. Враховуючи, що модулі вибираються більші за одержані частини повідомленння на виході блоку 2 ми отримуємо повідомлення в системі залишкових класів у вибраній системі модулів.

Рисунок 2.1 – Алгоритм формування перевірочних символів

В блоці 3 здійснюється перетворення повідомлення в позиційну систему числення з якого в блоці 4 обчислюються залишки по перевірочних модулях. В блоці 5 формується пакет, який складається з повідомлення та обчислених перевірочних символів.

Суттєвою перевагою розробленого алгоритму є те, що повідомлення залишається в позиційній системі числення.

3 РЕАЛІЗАЦІЯ ТА ДОСЛІДЖЕННЯ АЛГОРИТМІВ ОБЧИСЛЕННЯ ПЕРЕВІРОЧНИХ СИМВОЛІВ

3.1 Пристрій обчислення перевірочних символів в коригуючих кодах системи залишкових класів

Підвищення швидкодії апаратних засобів обробки даних можна досягти завдяки ефективним методам і алгоритмам обробки та новій технології виготовлення апаратних засобів, зокрема, використанню апаратних засобів з вищою тактовою частотою. Задача підвищення швидкодії особливо актуальна при обробці мультимедійних даних, які характеризуються великими обсягами та чутливістю до затримки, а також при обробці даних в протоколах передачі телекомунікаційних мереж, зокрема, при застосуванні коректуючих кодів. Оскільки обробка даних часто пов’язана з виконанням великої кількості арифметичних операцій, то від ефективності реалізації останніх в значній мірі залежить ефективність обробки в цілому. Як відомо, виконання арифметичних операцій в системі залишкових класів (СЗК) має ряд переваг, які досягаються за рахунок можливості паралельного виконанням арифметичних операцій, малої (3-6 біт) розрядності залишків та незалежності залишків [48 - 52].

Основною елементною базою для реалізації систем обробки даних на даний час є мікропроцесори, мікроконтролери, програмовані логічні інтегральні схеми (ПЛІС) та системи на кристалі (System-on-a-Chip, SoC). Ефективність виконання арифметичних операцій в СЗК на ПЛІС в значній мірі залежить від методів та алгоритмів їх виконання. В дисертаційній роботі у якості критеріїв оцінки ефективності виконання арифметичних операцій вибрано апаратні затрати (кількість логічних елементів) та час виконання відповідних операцій (максимальна затримка проходження сигналу) .

Для ефективної реалізації компонентів БСМ на основі запропонованих в попередніх розділах методів необхідно дослідити методи виконання арифметичних операцій в модулярній арифметиці при їх реалізації на ПЛІС []49-51.

На основі запропонованого в пункті 2.3 алгоритму формування коригуючих кодів СЗК, розроблено структуру та реалізовано на ПЛІС пристрій кодування.

Процес кодування складається із трьох етапів (рисунок 3.1):

1) розділення вхідного повідомлення на k частин (4 – 8 біт);

2) переведення повідомлення в позиційну систему числення в вибраній системі модулів;

3) обчислення перевірочних символів.

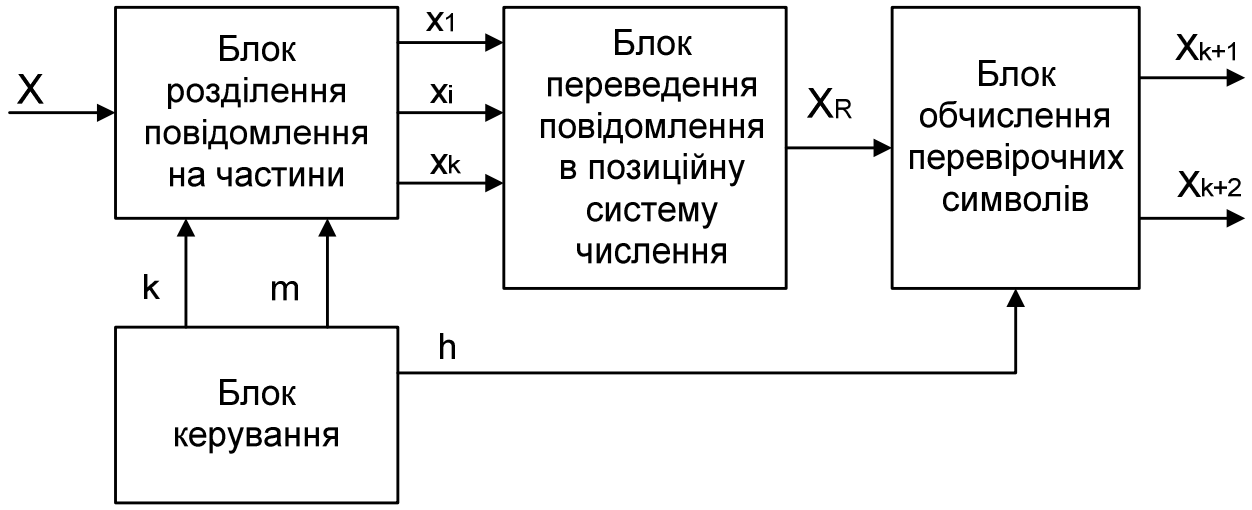

Рисунок 3.1 – Узагальнена структура пристрою кодування

Процес розділення вхідного повідомлення полягає в присвоєнні змінним x_1, x_2, … x_i, … x_k m - біт повідомлення X, де k – кількість інформаційних модулів.

При $m=8$ і $k=4$ процес розділення на мові Verilog має вигляд:

```
input [31:0] X; # вхідне повідомлення
assign  x1 = X [7:0];
assign  x2 = X [15:8];
assign  x3 = X [23:16];
assign  x4 = X [31:24].
```

При синтезі на ПЛІС описана операція не вимагає апаратних затрат.

В якості інформаційних модулів вибрано прості числа: p1=257, p2=263, p3=269, p4=271. Перевірочні модулі: p5=277, p6=281.

Базисні числа рівні:

B1 = 3201796979,

B2 = 393435903,

B3 = 3113917370,

B4 = 3145482367.

Робочий діапазон P = 4927316309.

Процедура обчислення позиційного значення повідомлення:

```
assign A1 = (x1*B1)%P;
assign A2 = (x2*B2)%P;
assign A3 = (x3*B3)%P;
assign A4 = (x4*B4)%P;
assign XR = (A1+A2+A3+A4)%P.
```

В блоці обчислення перевірочних символів відбувається обчислення залишків за перевірочними модулями (див. рисунок 3.2):

```
assign x_5 = XR %p5;
assign x_6 = XR %p6.
```

Для розширення функціональних можливостей кодера в блоці керування задається розрядність повідомлення – m, кількість інформаційних модулів – k та кількість перевірочних модулів – h (див. рисунок 3.10). На виходах пристрою $x_{k+1}[8:0]$, $x_{k+2}[8:0]$ формуються перевірочні символи в залежності від вхідного значення X_R (h – вказує кількість перевірочних модулів 1 або 2) та

перевірочних модулів. Розрядність вхідного повідомлення задається кодом на вході k відповідно до таблиці 3.1.

Таблиця 3.1 – Задання розрядності вхідного повідомлення

Код на вході k	Розрядність вхідного повідомлення, біт
001	16
010	24
011	32
100	40
101	48

Роботу кодера описано на мові Verilog. Моделювання та синтез кодера виконано в середовищі Quartus II (рисунок 3.2). Пристрій реалізовано на ПЛІС фірми «Altera», серія Cyclone IV. Функціональна схема кодера приведена на рисунку 3.3.

Для перевірки правильності роботи кодера вхідне повідомлення X та вихідні значення x_1, x_2, x_3, x_4 на часовій діаграмі подані в двійковій системі числення, значення A32 отримане по залишках (x_1, x_2, x_3, x_4) і перевірочні символи x_5 і x_6 подані в десятковій системі числення (рисунок 3.2):

$$x\_5 = A32(\mathrm{mod}\, p5) = 2260348044(\mathrm{mod}\, 277) = 67;$$

$$x\_6 = A32(\mathrm{mod}\, p6) = 2260348044(\mathrm{mod}\, 281) = 61.$$

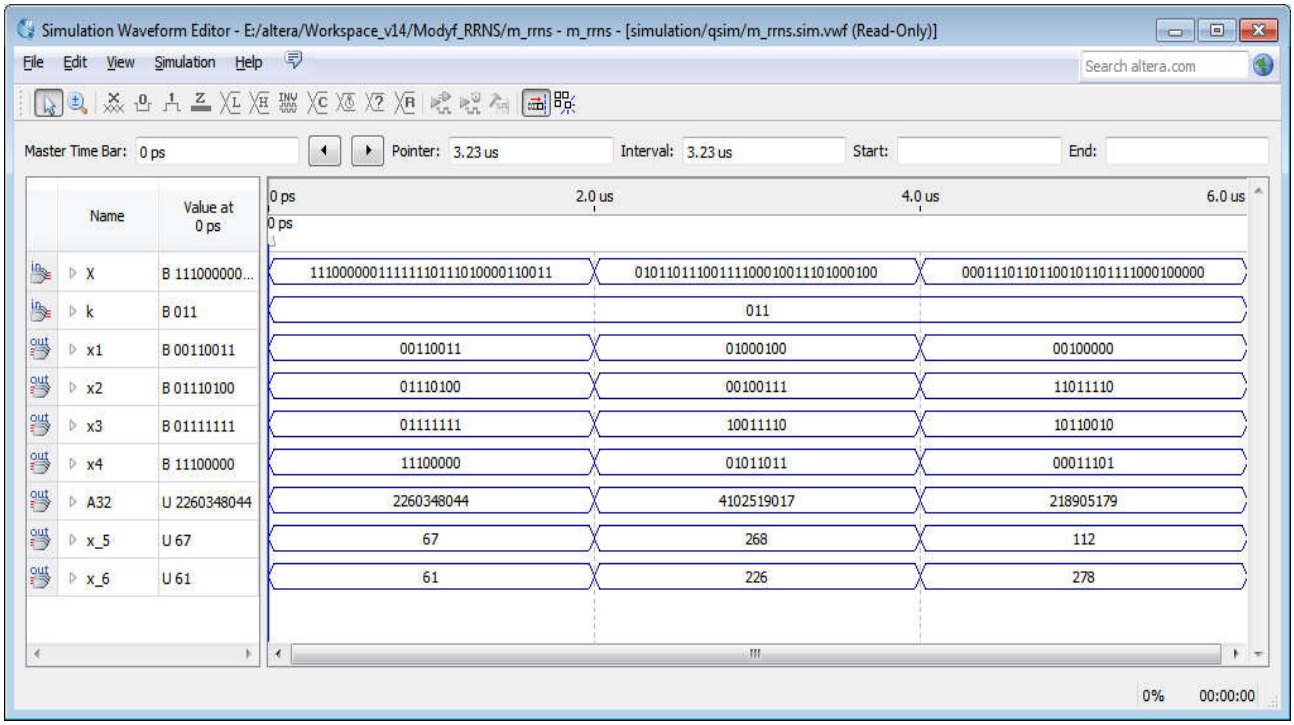

Рисунок 3.2 – Результати симуляції роботи кодера при розрядності вхідних даних n=32

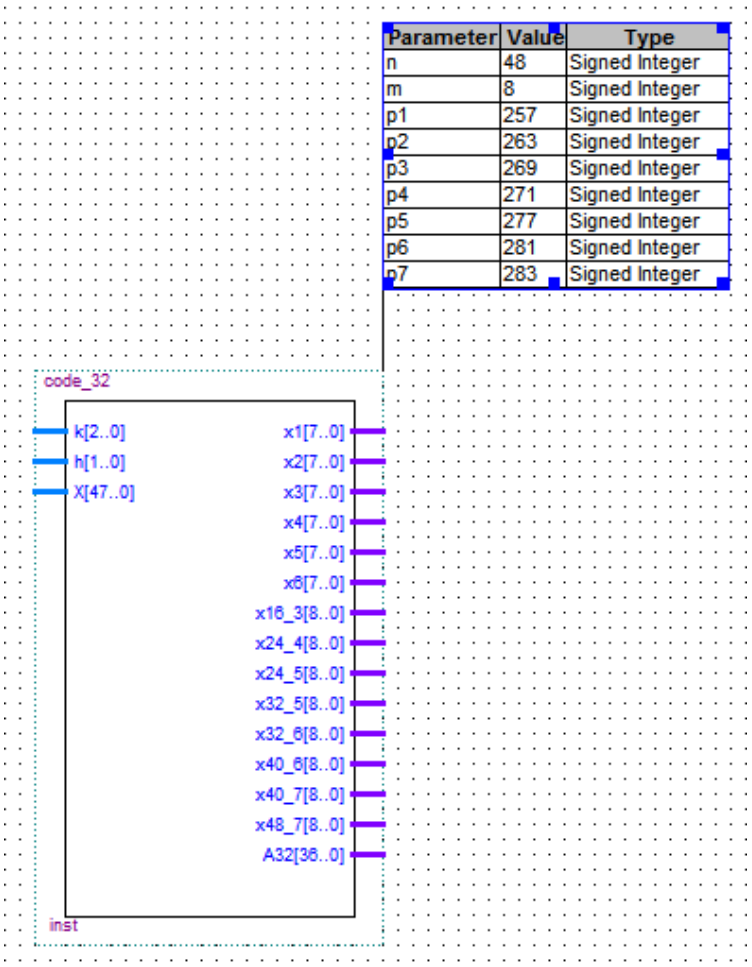

Рисунок 3.3 – Інтерфейс кодера

Інтерфейс кодера наведено на рисунку 4.10. При $h = 0$ – обчислюється один перевірочний символ, при $h = 1$ – два перевірочні символи. Вхід k визначає розрядність вхідного повідомлення, відповідно до таблиці 3.2.

Апаратні затрати (кількість логічних елементів) та максимальна затримка сигналу при реалізації кодера на ПЛІС для різної розрядності вхідних даних приведені в таблиці 3.2.

Таблиця 3.2 – Параметри синтезу кодера для різної розрядності повідомлення

Розрядність повідомлення	16	24	32	40	48
Всього логічних елементів	1,113 / 21,280 (5 %)	1,846 / 21,280 (9 %)	2,472 / 21,280 (12 %)	3,127 / 21,280 (15 %)	3,745 / 21,280 (18 %)
Всього контактів	84 / 167 (50 %)	92 / 167 (55 %)	100 / 167 (60 %)	108 / 167 (65 %)	116 / 167 (69 %)
Час затримки, нс	71.663	101.822	118.132	148.625	187.592

Експериментально встановлено, що реалізація кодера на основі запропонованого методу формування коригуючих кодів СЗК в порівнянні з відомими методами, забезпечує зменшення апаратних затрат на 20% в залежності від розрядності повідомлення та підвищення швидкодії за рахунок відсутності перетворення повідомлення в СЗК.

3.2 Розробка модуля обчислення перевірочних символів

Підвищення швидкодії обчислення перевірочних символів, тобто обчислення залишків за перевірочними модулями забезпечується тим, що пристрій містить неповні шифратори (рисунок 3.4).

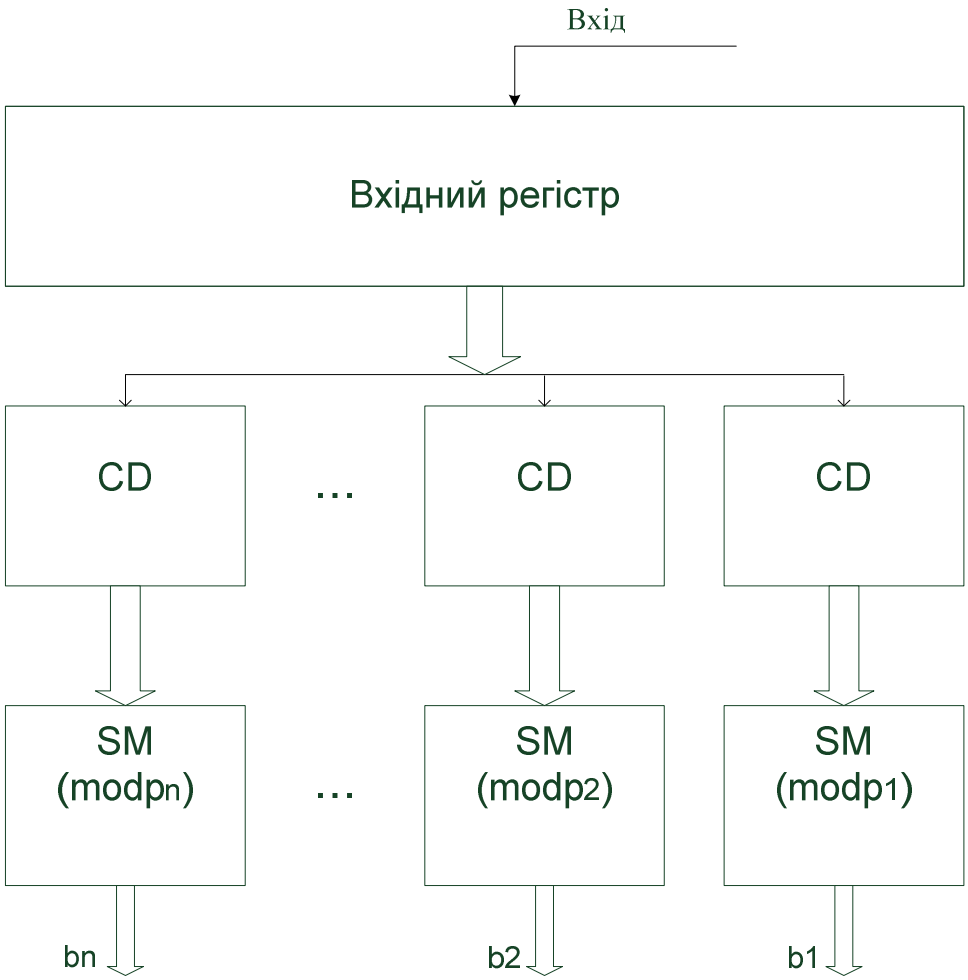

Рисунок 3.4 – Пристрій перетворення паралельного двійкового коду в код системи залишкових класів: 1 – вхідний паралельний регістр; 2 – неповні шифратори; 3 – пірамідальні суматори по заданому модулю.

Модуль містить вхідний *n* розрядний регістр зберігання даних, шифратори та суматори по відповідному модулю, виходи шифраторів з'єднані з входами суматорів по модулю, відсутня схема керування а залишки по заданому модулю від основи два у відповідному степені формуються в

неповних шифраторах, входи яких підключені до виходів регістра, виходи шифраторів підключені до входів пірамідального суматора по заданих модулях, код системи залишкових класів формується на виходах пірамідальних суматорів по відповідних модулях.

На рисунку 3.5 зображена структурна схема неповного шифратора: a_j – значення розряду двійкового числа; c_i – двійковий код коефіцієнтів. На рисунку 3.6 зображена структурна схема пірамідального суматора по модулю: 1 – суматор по модулю; c_i – значення i - го коефіцієнта; b_i – код системи залишкових класів.

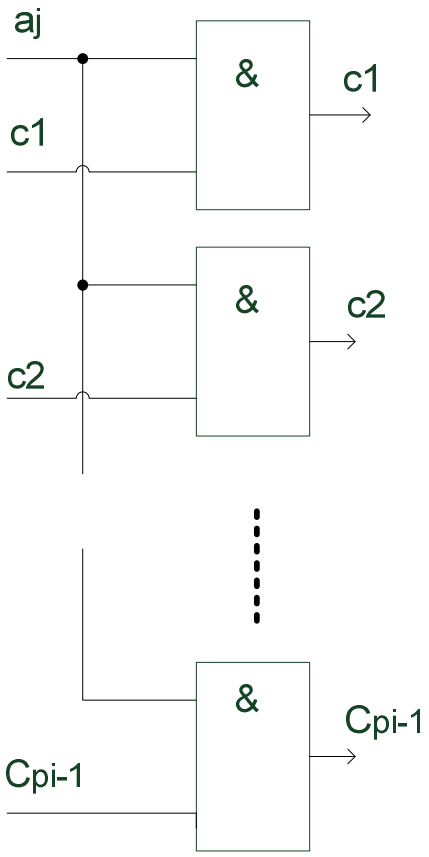

Рисунок 3.5 – Пристрій для перетворення паралельного двійкового коду в код системи залишкових класів

Модуль працює наступним чином. Двійковий код, який підлягає перетворенню поступає на вхідний регістр 1, з виходу регістра паралельний двійковий код поступає на входи неповних шифраторів (рисунок 3.6), при наявності в j розряді двійкового коду одиниці на виході шифратора 2

формується значення $c_{ij} = (2^j) \bmod p_i$, де p_i – модуль системи числення, $i = \overline{1, n}$, n – кількість модулів, з виходу неповних шифраторів значення c_{ij} поступають на вхід пірамідальних суматорів (рисунок 3.6), які працюють по модулях p_i. На виходах пірамідальних суматорів формується код системи залишкових класів по заданих модулях. Виходи пірамідальних суматорів 3 є виходами пристрою.

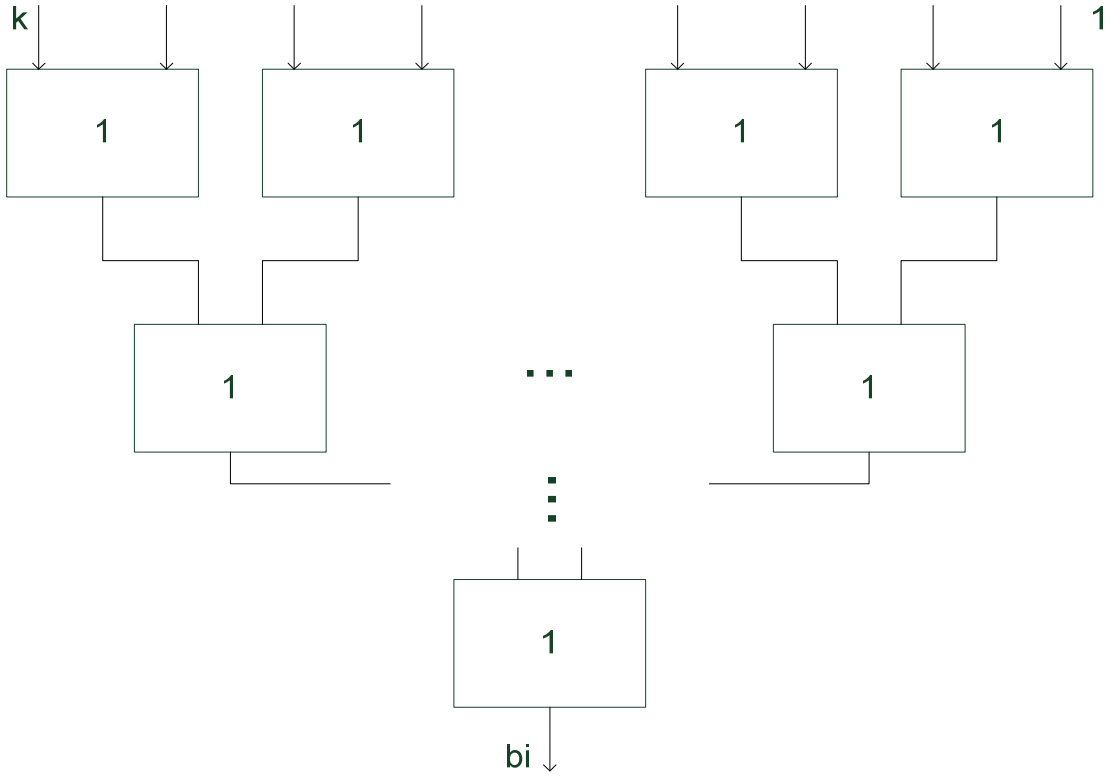

Рисунок 3.6 – Пірамідальний суматор по модулю

Робота шифраторів описується системою логічних рівнянь [28, 53, 54]:
– для модуля $p_1 = 7$ (рисунок 3.7):

$$f_{p1\_0}[0] = EO \wedge a_0;$$
$$f_{p1\_1}[1..0] = EO \wedge (a_1 \vee a_0);$$
$$f_{p1\_2}[2..0] = EO \wedge (a_2 \vee a_1 \vee a_0);$$
$$\dots$$
$$f_{p1\_23}[2..0] = EO \wedge (a_2 \vee a_1 \vee a_0);$$

– для модуля $p_2 = 23$:

$$f_{p2\_0}[0] = EO \wedge a_0;$$
$$f_{p2\_1}[1..0] = EO \wedge (a_1 \vee a_0);$$
$$f_{p2\_2}[2..0] = EO \wedge (a_2 \vee a_1 \vee a_0);$$
$$\dots$$
$$f_{p2\_6}[4..0] = EO \wedge (a_3 \vee a_2 \vee a_1 \vee a_0);$$
$$\dots$$
$$f_{p2\_23}[1..0] = EO \wedge (a_1 \vee a_0);$$

– для модуля $p_3 = 29$:

$$f_{p3\_0}[0] = EO \wedge a_0;$$
$$f_{p3\_1}[2..0] = EO \wedge (a_2 \vee a_1 \vee a_0);$$
$$\dots$$
$$f_{p3\_14}[4..0] = EO \wedge (a_4 \vee a_3 \vee a_2 \vee a_1 \vee a_0);$$
$$\dots$$
$$f_{p3\_23}[3..0] = EO \wedge (a_3 \vee a_2 \vee a_1 \vee a_0);$$

– для модуля $p_4 = 59$:

$$f_{p4\_0}[0] = EO \wedge a_0;$$
$$f_{p4\_1}[1..0] = EO \wedge (a_1 \vee a_0);$$
$$\dots$$
$$f_{p4\_23}[5..0] = EO \wedge (a_5 \vee a_4 \vee a_3 \vee a_2 \vee a_1 \vee a_0);$$

– для модуля $p_5 = 61$:

$$f_{p5\_0}[0] = EO \wedge a_0;$$
$$f_{p5\_1}[1..0] = EO \wedge (a_1 \vee a_0);$$

…

$$f_{p5\_23}[3..0] = EO \wedge (a_3 \vee a_2 \vee a_1 \vee a_0);$$

де EO – дозволяючий вхід, при $EO = 1$ на виході шифратора формується код.

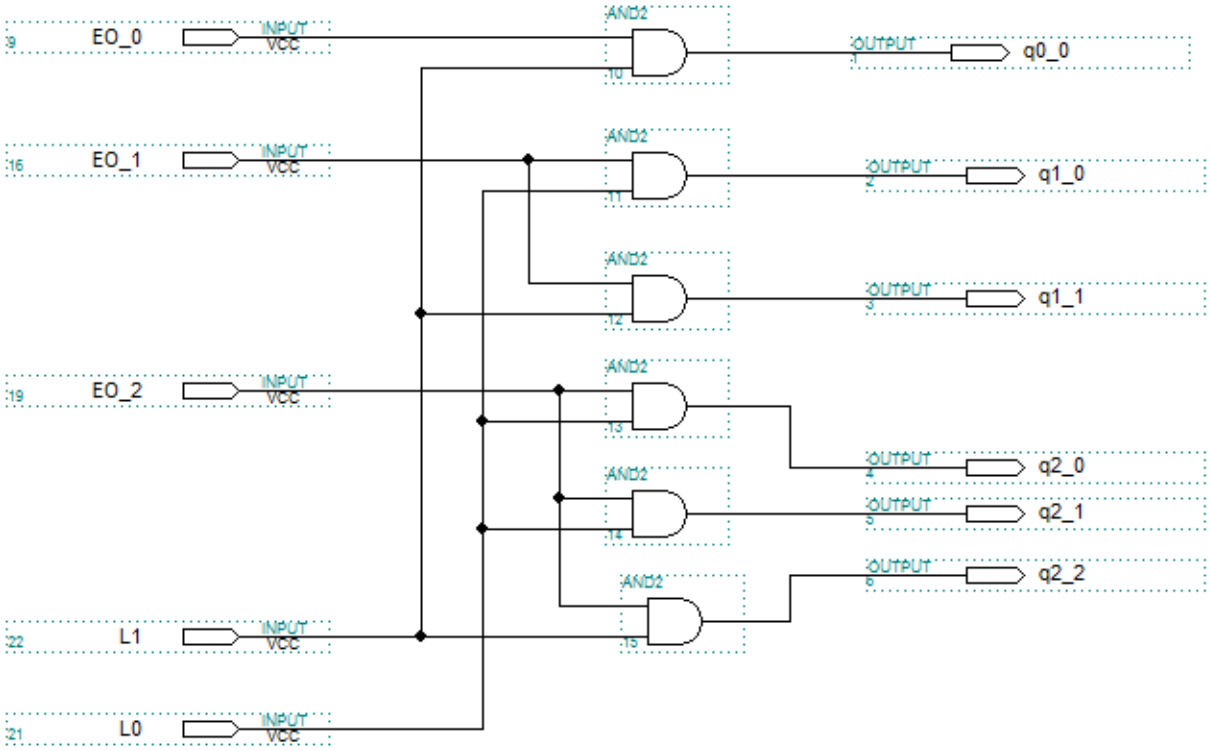

Рисунок 3.7 – Схема шифратора формування коефіцієнтів для модуля $p_1 = 7$

Робота сумматора по модулю $p_1 = 7$ описується таблицею істинності (таблиця 3.3).

Зі збільшенням розрядності модуля, значно збільшується складність запису логічних рівнянь, тому робота суматорів по модулях $p_1 = 7$, $p_2 = 23$, $p_3 = 29$, $p_4 = 59$, $p_5 = 61$, описана на VHDL (рисунок 3.8 – рисунок 3.12).

Таблица 3.3 – Таблиця істинності суматора по модулю 7

x_1	x_2						
	000	001	010	011	100	101	110
000	000	001	010	011	100	101	110
001	001	010	011	100	101	110	111
010	010	011	100	101	110	000	001
011	011	100	101	110	000	001	010
100	100	101	110	000	001	010	011
101	101	110	000	001	010	011	100
110	110	000	001	010	011	100	101

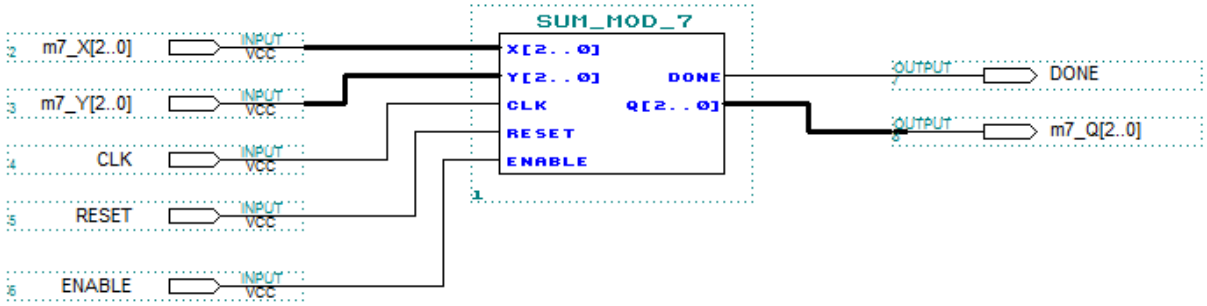

Рисунок 3.8 – Суматор по модулю 7.

Проведена верифікація розроблених модулів підтвердила їх коректну роботу (рисунок 3.9, 3.11).

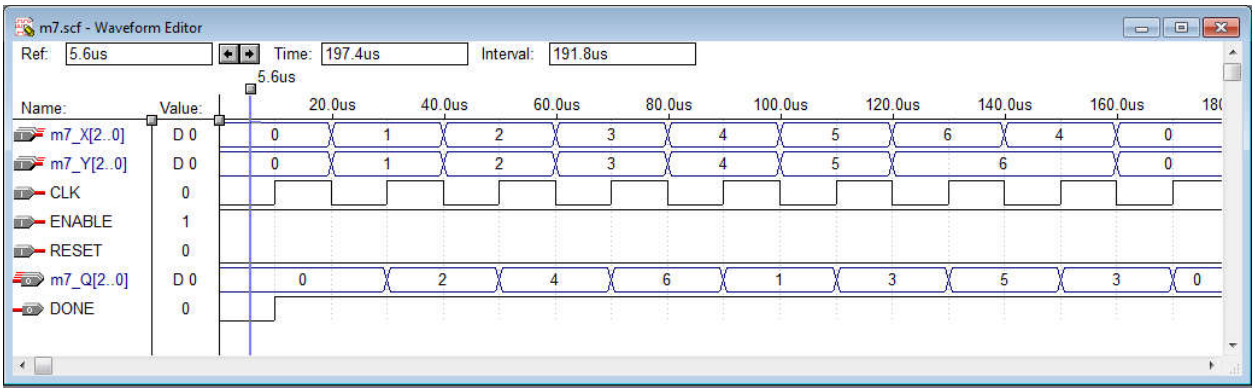

Рисунок 3.9 – Верифікація суматора по модулю 7

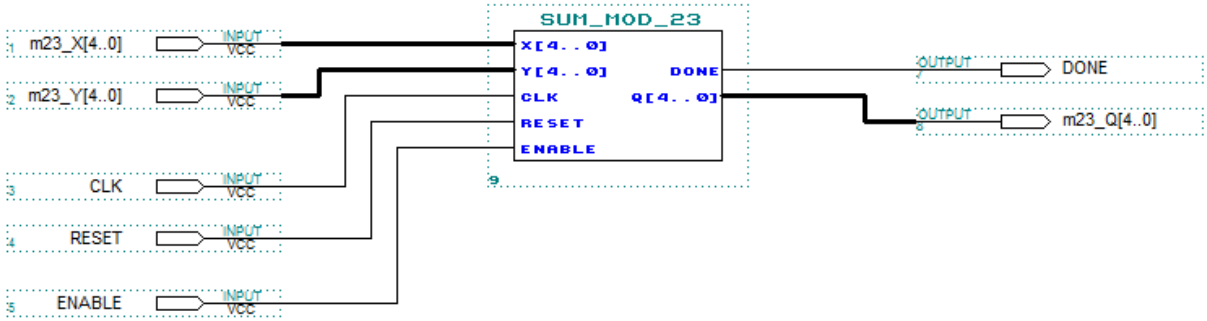

Рисунок 3.10 – Суматор по модулю 23

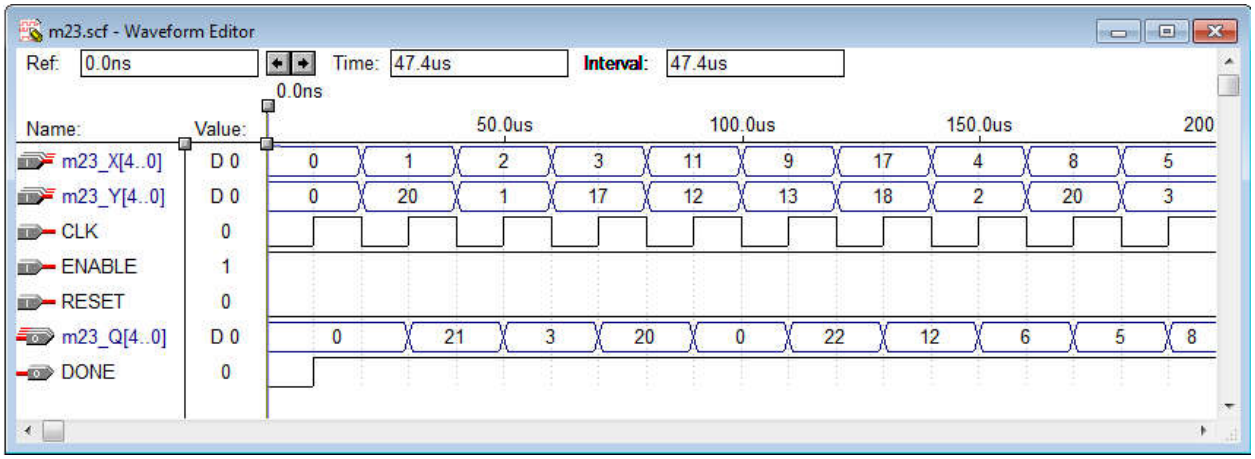

Рисунок 3.11 – Часові діаграми роботи суматора по модулю 23

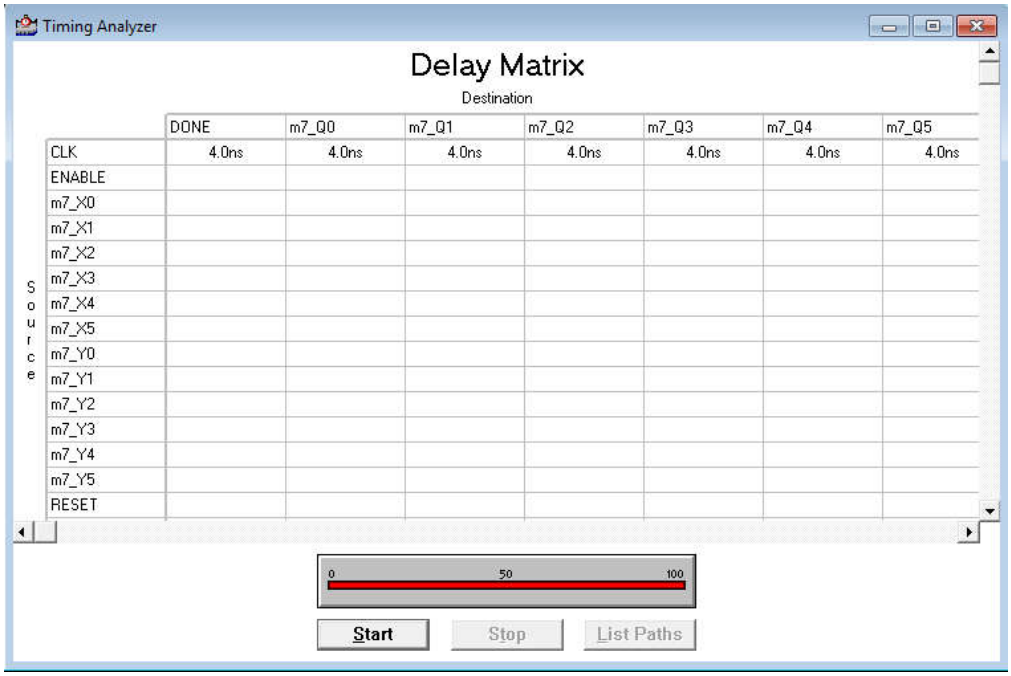

Рисунок 3.12 – Час отримання сигналу на виході суматора по модулю 61

Багаторозрядний суматор по модулю 7, реалізований в САПР Max + Plus II (рисунок 3.13).

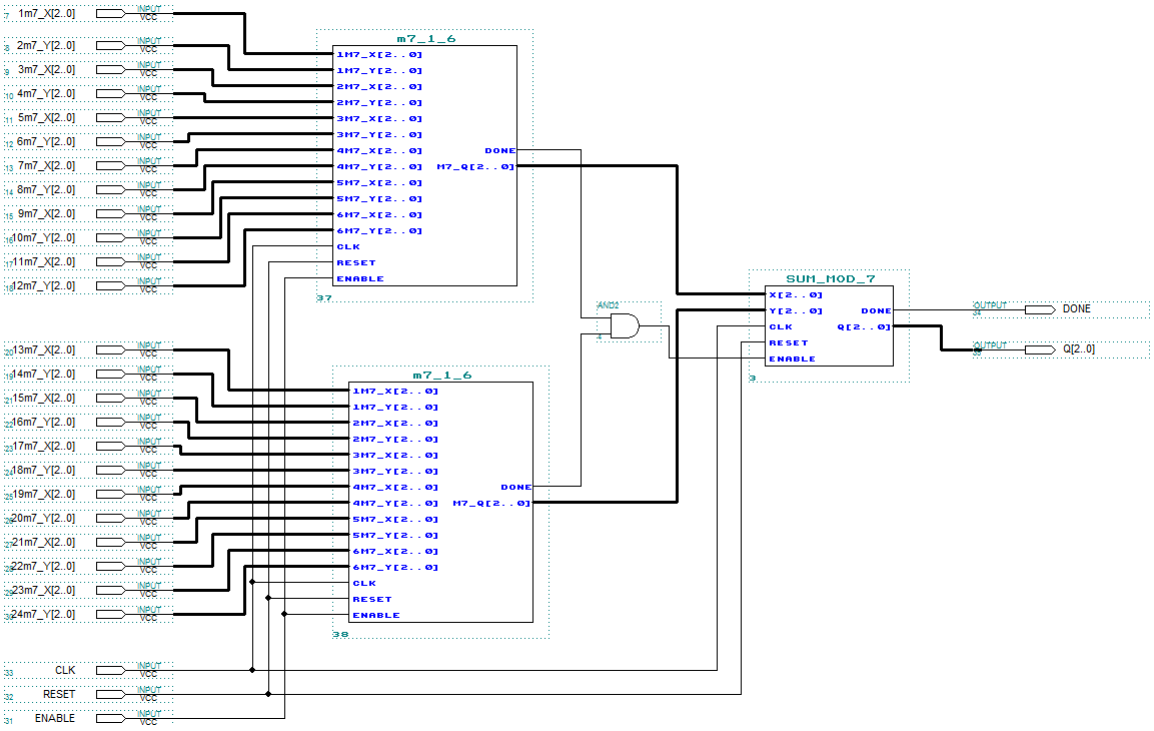

Рисунок 3.13 – Багаторозрядний суматор по модулю 7

Розроблений пристрій перетворення двійкового коду в систему залишкових класів забезпечує високу швидкодію за рахунок використання неповних шифраторів і паралельно-послідовного багаторозрядного суматора за відповідними модулями. Пристрій реалізований на CPLD фірми ALTERA серії MAX-II, мікросхема EPM240T100C5, час перетворення 24 - бітного двійкового коду становить 16,7 нс [55].

Розроблений пристрій реалізований на програмованій логічній інтегральній схемі (ПЛІС) і може бути використаний для введення даних представлених в двійковому коді в процесори оброблення інформації, які працюють в системі залишкових класів і забезпечують високу швидкодію виконання арифметичних операцій в системах реального часу.

3.3 Експерементальніх дослідження пристрою обчислення перевірочних символів

В даному розділі представленні результати експерементальних досліджень апаратних затрат (кількість логічних елементів) та часу обчислення перевірочних символів при різній розрядності вхідних даних, для коректуючих кодів СЗК та для запропонованого методу формування коректуючих кодів СЗК. Експерементальні дослідження проводилися з використанням програмного забесперченням Quartus фірми Intel (Altera). Відомий та запропонований метод колування коректуючих кодів СЗК, описані на мові програмування апаратних засобів Verilog-HDL, та синтезовані в мікросхемах Cyclone IV.

Для проведення експерементів вибрані такі дані:

– розрядність вхідних даних від 16 до 48 біт;
– кількість інформаційних модулів - 4;
– кількість перевірочних модулів - 1, 2;
– значення модулів залежить від розрядності повідомлення.

Дослідження запропонованого методу формування перевірочних символів коректуючих кодів СЗК (метод 2) проведено з відомими коректуючими кодами СЗК (метод 1) [28].

На рисунку 3.14, 3.15 приведено результати експерементальних досліджень апаратних затрат (кількість логічних елементів) та часу обчислення перевірочних символів при різній розрядності вхідних даних, для коректуючих кодів СЗК та для запропонованого методу з одним перевірочним модулем.

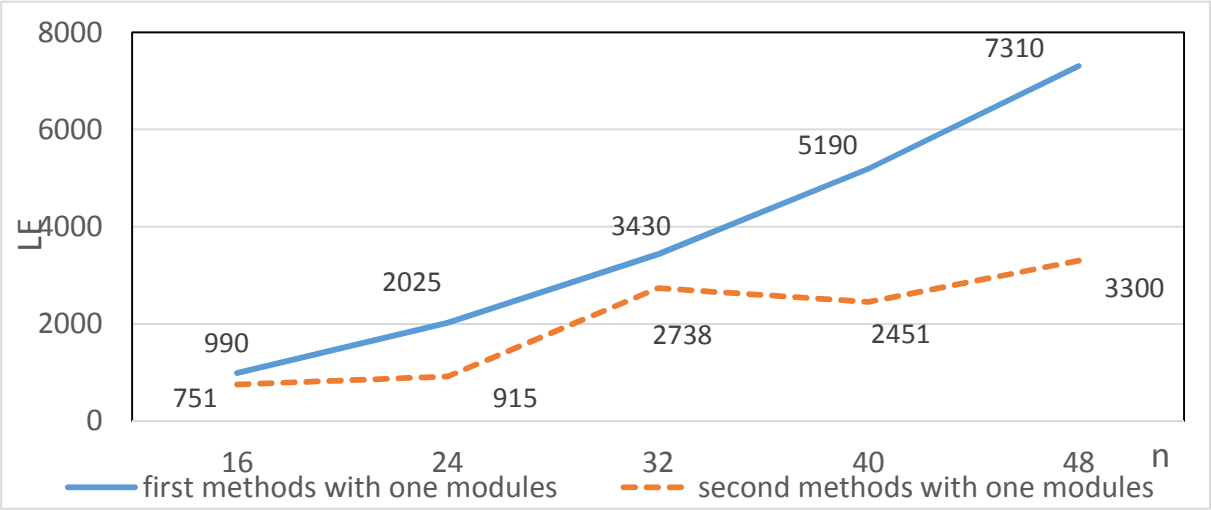

Рисунок 3.14 – Залежність апаратних затрат від розрядності вхідних даних при одному перевірочному модулі для методу 1 і 2

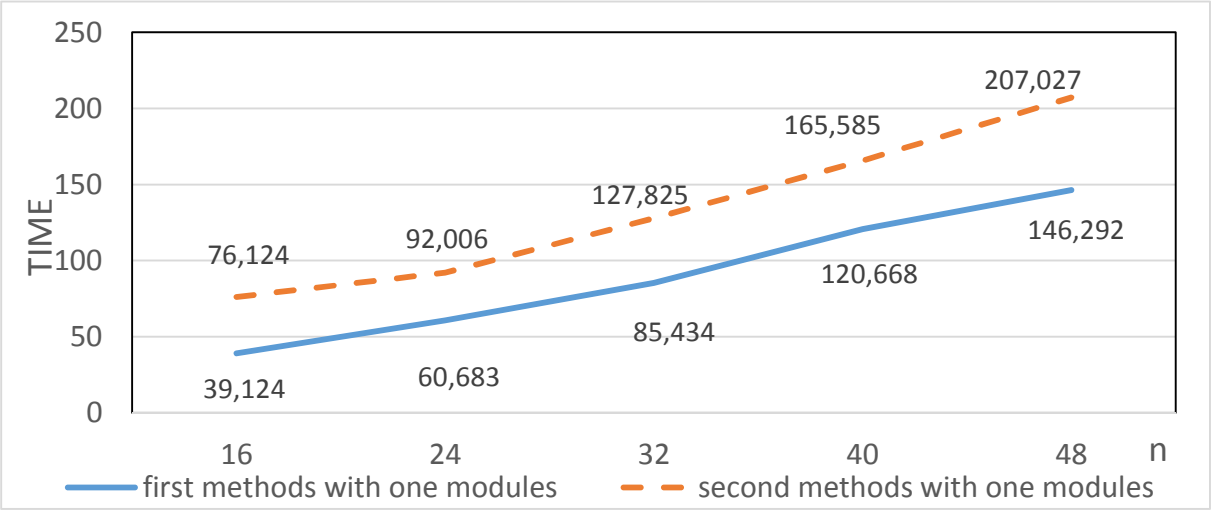

Рисунок 3.15 – Залежність часу обчислення перевірочних символів від розрядності вхідних даних при одному перевірочному модулі для методу 1 і 2.

Як видно з рисунку 3.14 запропонований метод забезпечує зменшення апаратних затрат в середньому на 35% в залежності від розрядності вхідних даних. При цьому часові затрати зростають в середньому на 46% (рисунок 3.15).

На рисунку 3.16, 3.17 приведено результати експерементальних досліджень апаратних затрат (кількість логічних елементів) та часу обчислення перевірочних символів при різній розрядності вхідних даних, для коректуючих кодів СЗК та для запропонованого методу з двома перевірочними модулями.

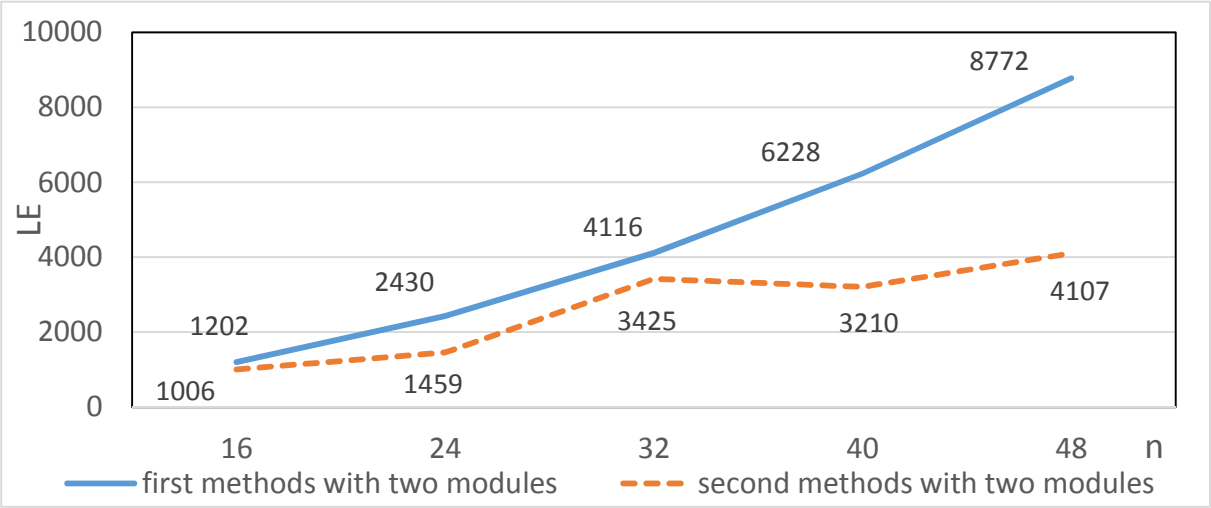

Рисунок 3.16 – Залежність апаратних затрат від розрядності вхідних даних при двох перевірочних модулях для методу 1 і 2.

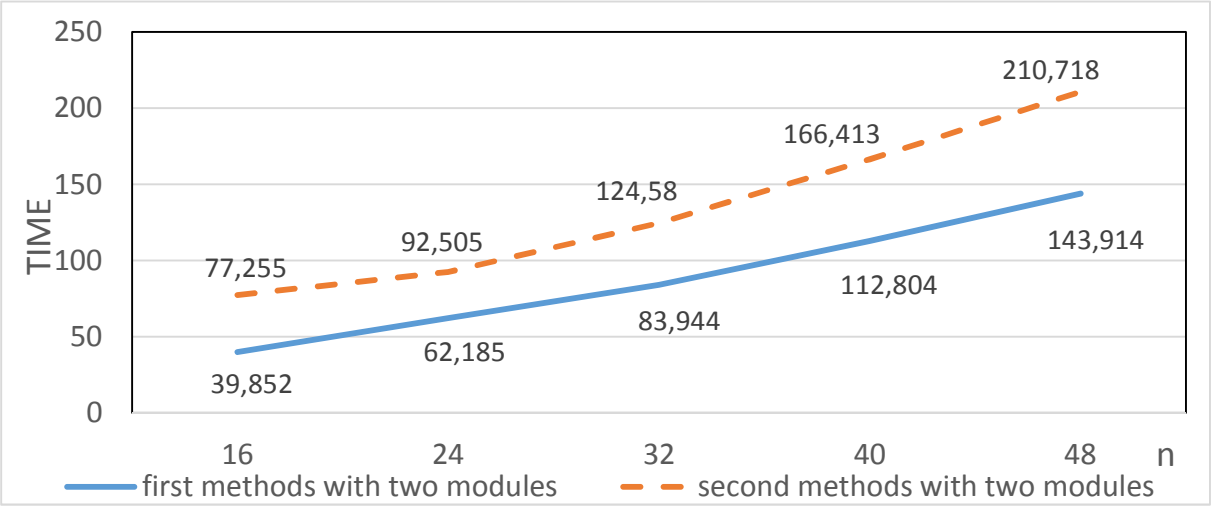

Рисунок 3.17 – Залежність часу обчислення перевірочних символів від розрядності вхідних даних при двох перевірочних модулях для методу 1 і 2.

Як видно з рисунку 3.16 запропонований метод забезпечує зменшення апаратних затрат в середньому на 29% в залежності від розрядності вхідних даних. При цьому часові затрати зростають в середньому на 47% (рисунок 3.17).

Також проведено досдіження апаратних затрат (рисунок 3.18) та швидкодії обчислення (рисунок 3.19) при поділі вхідних даних на 8 біт. При цьому значення всіх модуів є більші за 256.

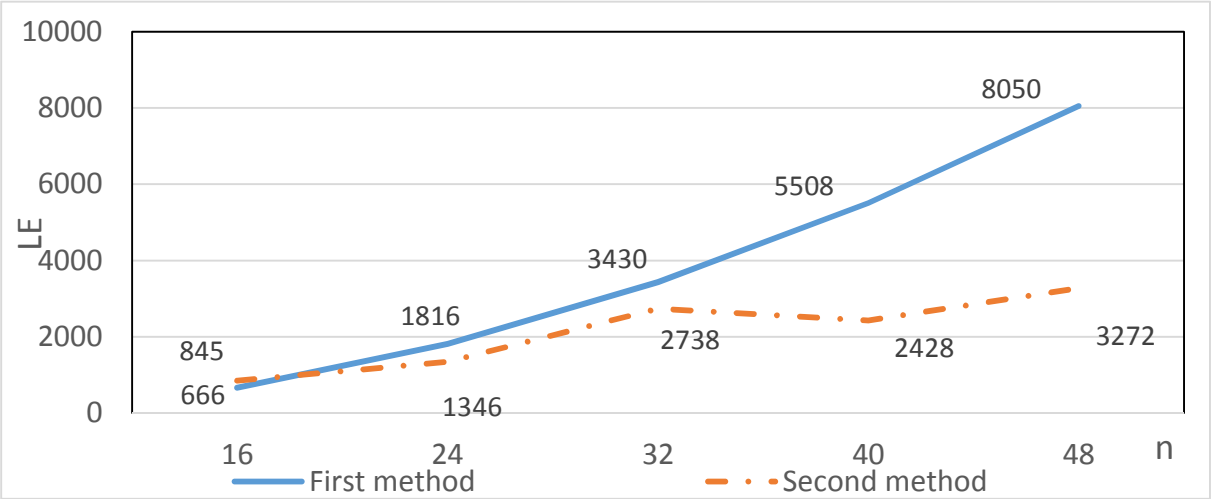

Рисунок 3.18 – Залежність апаратних затрат від розрядності вхідних даних при одному перевірочному модулі для методу 1 і 2 (розрядність блоку 8 біт).

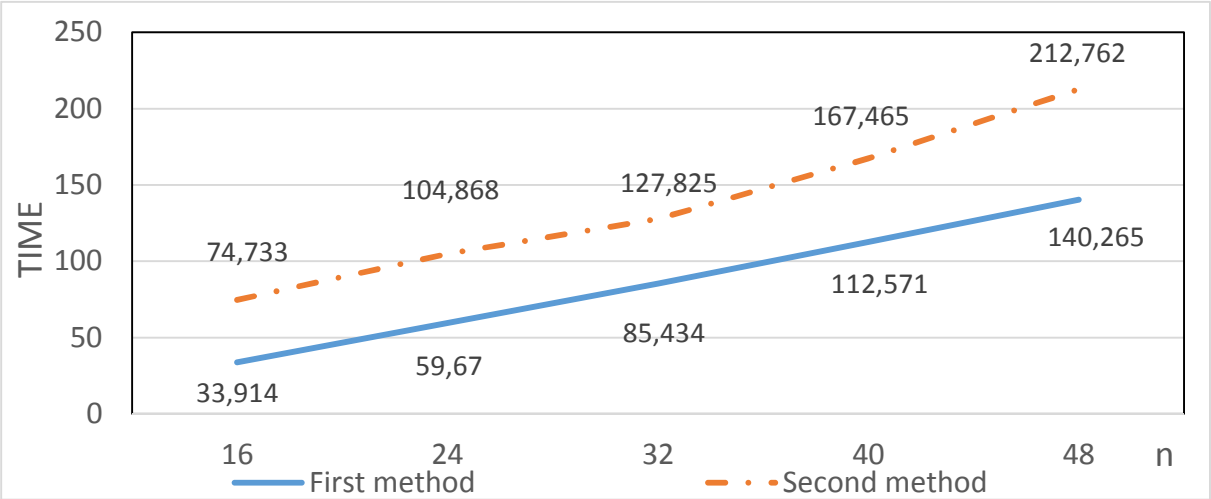

Рисунок 3.19 – Залежність часу обчислення перевірочних символів від розрядності вхідних даних при одному перевірочному модулі для методу 1 і 2 (розрядність блоку 8 біт).

Як видно з рисунку 3.18 запропонований метод забезпечує зменшення апаратних затрат в середньому на 23% і залежить від розрядності вхідних даних (розрядність блоку 8 біт). При цьому часові затрати зростають в середньому на 58% (рисунок 3.19).

Проведено досдіження апаратних затрат (рисунок 3.20) та швидкодії обчислення (рисунок 3.21) при поділі вхідних даних на 8 біт та двох перевірочних модулях. При цьому значення всіх модуів є більші за 256.

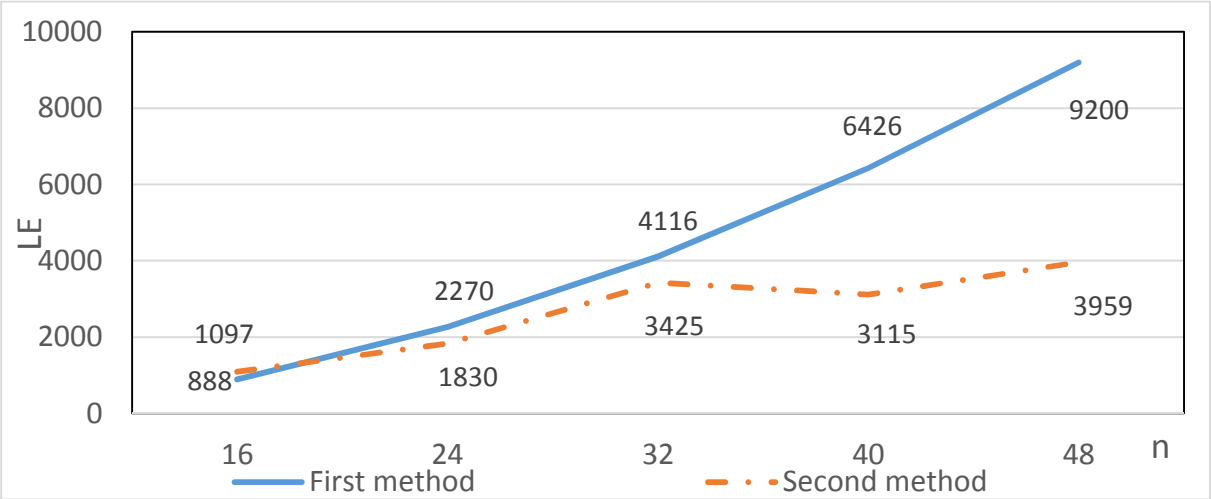

Рисунок 3.20 – Залежність апаратних затрат від розрядності вхідних даних при двох перевірочних модулях для методу 1 і 2 (розрядність блоку 8 біт).

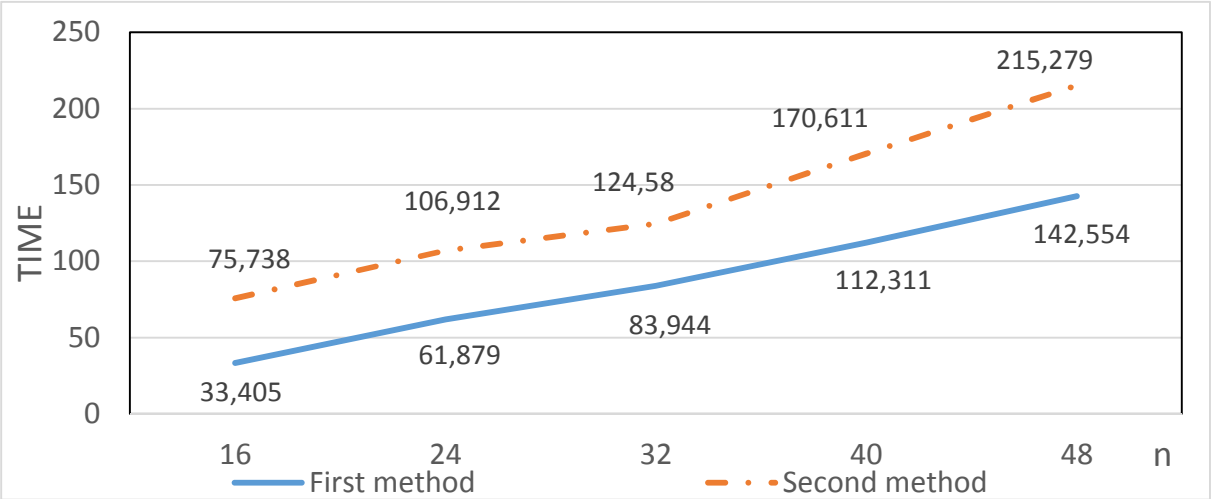

Рисунок 3.21 – Залежність часу обчислення перевірочних символів від розрядності вхідних даних при двох перевірочних модулях для методу 1 і 2 (розрядність блоку 8 біт).

Як видно з рисунку 3.18 розроблений метод забезпечує зменшення апаратних затрат в середньому на 20% і залежить від розрядності вхідних даних (розрядність блоку даних 8 біт). Часові затрати на формування перевірочних символів в запропонованому методі зростають в середньому на 59% (рисунок 3.19) порівняно з відомим методом. Однак у відомому методі повідомлення знаходиться в системі залишкових класів, що потребує додаткового часу на зворотне перетворення в позиційну систему числення.

Розроблений алгоритм формування коригуючих кодів системи залишкових класів забезпечує зменшення апаратних затрат в середньому на 20% в залежності від розрядності повідомлення. При цьому, основною перевагою запропонованого методу є те, що вхідне повідомлення обробляється в позиційній системі числення тобто не потребує перетворення в систему залишкових класів. Таким чином розроблений алгоритм формування перевірочних символів значно розширить область застосування корегувальних кодів системи залишкових класів за рахунок обробки повідомлень, які представлені в позиційних системах числення.

ВИСНОВКИ

Робота присвячена розв’язанню наукової задачі розробки та дослідження алгоритмів формування перевірочних символів в коригуючих кодах системи залишкових. Головний акцент було здійснено на методах переведення даних з позиційної системи числення в систему залишкових класів і зворотного перетворення.

1. Проведено аналіз та дослідження існуючих методів переведення даних з позиційної системи числення в систему залишкових класів, здійснено їх практичну реалізацію на програмованих логічних матрицях. Також було розроблено та реалізовано ряд модифікованих і нових методів прямого переведення даних.

2. Розроблено алгоритм формування коригуючих кодів системи залишкових класів основною перевагою якого є те, що вхідне повідомлення обробляється в позиційній системі числення тобто не потребує перетворення в систему залишкових класів, що забезпечує зменшення апаратних затрат.

3. Експериментально встановлено, що реалізація кодера на основі запропонованого методу формування коригуючих кодів системи залишкових класів в порівнянні з відомими методами, забезпечує зменшення апаратних затрат на 20% в залежності від розрядності повідомлення та підвищення швидкодії за рахунок відсутності перетворення повідомлення в систему залишкових класів.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Goh, Vik Tor, Mohammad Umar Siddiqi. Multiple error detection and correction based on redundant residue number systems. Communications, *IEEE* Transactions. – 2008. – №56.3. – P.325-330

2. Hu Zhengbing. Increasing the Data Transmission Robustness in WSN Using the Modified Error Correction Codes on Residue Number System /Hu Zhengbing, V. Yatskiv, A. Sachenko // Elektronika ir Elektrotechnika, 2015. –Vol 21. – № 1. – P. 76-81.

3. Krasnobayev, V., Yanko, A., Koshman, S.: A Method for arithmetic comparison of data represented in a residue number system. Cybern. Syst. Anal. 52(1), 2016. – P. 145–150.

4. Omondi A. Residue Number System: Theory and Implementation / A.Omond, B.Premkumar. Imperial College Press, 2007. – Vol. 2. – 296 p.

5. Roshanzadeh M., Saqaeeyan S. Error Detection & Correction in Wireless Sensor Networks By Using Residue Number Systems. International Journal of Computer Network and Information Security (IJCNIS) 4.2, 2012.– №2. – P. 29-35.

6. Su Jun. Method and Device for Image Coding & Transferring Based on Residue Number System / Su Jun, V.Yatskiv // Sensors & Transducers Journal, 2013. – Vol.18, Special Issue. – P.60-65.

7. Tay Thian Fatt. A new algorithm for single residue digit error correction in Redundant Residue Number System / Tay Thian Fatt, Chang Chip-Hong // Circuits and Systems (ISCAS), IEEE International Symposium IEEE, 2014. – P. 1748-1751.

8. Tay Thian Fatt. A new algorithm for single residue digit error correction in Redundant Residue Number System / Tay Thian Fatt, Chang Chip-Hong // Circuits and Systems (ISCAS), IEEE International Symposium IEEE, 2014. – P. 1748-1751.

9. Wang Y. Residue-to-Binary Converters Based On New Chinese Remainder Theorems. IEEE Transactions on Circuits and Systems – II: Analog and Digital Signal Processing, 2000 – Vol. 47, No. 3. – P.197–205.

10. Yatskiv V. CPLD Encoder and Decoder for Modified Correction Codes Based on Residue Number System // V.Yatskiv, N. Yatskiv, A. Sachenko, Su Jun / Modern Problems of Radio Engineering, Telecommunications, and Computer Science. Proceedings of the International Conference TCSET'2014 Lviv-Slavske, Ukraine February 25 – March 1, 2014 – P. 492-493.

11. Yatskiv V. Error Correction Technique Based on Modular Correcting Codes /V. Yatskiv, T. Tsavolyk, A. Sachenko // Conference Proceedings IEEE 36th International Conference on Electronics and Nanotechnology (ELNANO), April 19-21, Kyiv, Ukraine, 2016 – P.362-364.

12. Yatskiv V. The network coding method in wireless sensor networks based on residue number system. Zeszyty naukowe politechniki śląskiej. Seria: Organizacja i Zarządzanie z. 67. 2013. – P. 135-144.

13. Yatskiv V., Tsavolyk T., Yatskiv N. The Correcting Codes Formation Method Based on the Residue Number System. Conference Proceedings of 14 th International Conference The Experience of Designing and Application of CAD Systems in Microelectronics (CADSM-2017) 21-25 February 2017 Polyana-Svalyava (Zakarpattya), Ukraine, 2017. – P. 237-240.

14. Акритас А. Основы компьютерной алгебры с приложениями: Пер.с анг.- М., Мир, 1994. - 544 с.

15. Акушский И.Я. Машинная арифметика в остаточных класах / И.Я.Акушский, Д.И.Юдицкий. – М.: Сов. Радио, 1968. – 460 с.

16. Акушский И.Я., Пак И.Т. Вопросы помехоустойчивого кодирования в непозиционном коде // Вопросы кибернетики. – 1977. – Т.28. – С.36-56.

17. Акушский И.Я., Юдицкий Д.И. Машинная арифметика в остаточных классах. – М. : Сов. радио, 1968. – 440 с.

18. Алгулиев Р.М. Сенсорные сети: состояние, решения и перспективы. /Р.М.Алгулиев, Т.Х.Фаталиев, Б.С.Агаев, Т.С.Алиев // Телекоммуникации. Ежемесячный научно-технический информационно-аналитический и учебно-методический журнал. – 2007. – №4. – С. 27–32.

19. Амато Вито. Основы организации сетей Cisco, том 1. Пер. с англ. – М.: Издательский дом "Вильяме", 2004. – 512 с.

20. Барсов В.И., Сорока Л.С., КраснобаевВ.А. Методология параллельной обработки информации в модулярной системе счисления: Монография.- Х.: МОН, УИПА, 2009. – 268 с.

21. Варгаузин В. Помехоустойчивое кодирование в пакетных сетях // ТелеМультиМедиа. – 2005. – №9. – С.10-16.

22. Гераїмчук М.Д., Івахів О.В., Паламар М.І., Шевчук Б.М. Основи побудови перспективних безпроводових сенсорних мереж. Монографія. – К.: ЕКМО, 2010. – 124 с.

23. Дунець Р.Б., Тиранський Д.Я. Дослідження часткової ре конфігурації ПЛІС // Радіоелектронні і комп'ютерні системи. – 2009. – №6(40). – С.240-244.

24. Дунець Р.Б., Тиранський Д.Я. Проблеми побудови частково реконфігурованих систем на ПЛІС // Радіоелектронні і комп'ютерні системи. – 2010. – №7(48). – С.200 – 204.

25. Евстигнеев В.Г., Сведо - Швец А.А., Краснобаев В.А. Арифметические алгоритмы для **q** - ичной системы счисления // АСУ летательных аппаратов. – Харьков : ХАИ. – 1982. – Вып. 4. – С. 165 - 168.

26. Жуков И.А., Дрововозов В.И., Способы повышения надежности и безопасности сбора информации в системах управления реального времени // Проблеми інформатизації та управління. –2008. – № 1(23). – С. 262– 276.

27. Зюко А.Г. Помехоустойчивость и эффективность систем передачи информации. – М.: Радио и связь, 1985. – 272 с.

28. Казимир В. В. Проектування комп'ютерних систем на основі мікросхем програмованої логіки: монографія / С. А. Іванець, Ю. О. Зубань, В. В. Казимир, В. В. Литвинов. – Суми : Сумський державний університет, 2013. – 313 с.

29. Коды, исправляющие ошибки. Питерсон У., Уэлдон Э.: Мир, 1976. - 593 с.

30. Коляда А.А., Пак И.Т. Модулярные структуры конвейерной обработки цифровой информации. – Минск: Наука, 1992. – 256 с.

31. Кондратенко Ю.П., Мохор В.В., Сидоренко С.А. Verilog-HDL для моделирования и синтеза цифрових электронных схем. Учебное пособие. Под редакцией Ю.П. Кондратенко. – Николаев.: Издательство НГГУ им. Петра Могилы, 2002. – 221 с.

32. Краснобаев В. А. Метод обработки данных в классе вычетов / Краснобаев В. А., Янко А. С., Кошман С. А., Сомов С. А., Бендес Ю. П. // Збірник наукових праць Харківського університету Повітряних сил. – 2014. – №2 – С.121-126.

33. Краснобаев В.А. Метод коррекции ошибок в СОК // АСУ и приборы автоматики. – 1987. – Вып. 82.– С. 112 - 115.

34. Краснобаев В.А. Методы повышения надежности специализированных ЭВМ систем и средств связи. – МО СССР, 1990. – 173 с.

35. Краснобаев В.А. Надежностная модель ЭВМ в системе остаточных классов // Электронное моделирование. – 1985. – № 4. – С. 44 - 46.

36. Краснобаев В.А. Техническая реализация метода коррекции ошибок в СОК // АСУ и приборы автоматики. – 1987. – Вып. 81. – С. 97 - 101.

37. Краснобаев В.А., Ирхин В.П. Вариант решения обратной задачи оптимального резервирования в системе остаточных классов // Кибернетика. – 1990. – № 3. – С. 123 - 125.

38. Краснобаєв В.А. Метод підвищення достовірності контролю даних, представлених у системі залишкових класів / В.А.Краснобаєв, С.О.Кошман, М.О.Мавріна // Кибернетика и системний аналіз. – 2014. – Том 50, №6 .– С.167 –175.

39. Методичні рекомендації до виконання магістерської роботи з освітнього ступеня “Магістр”. Спеціальність: 123 - Комп’ютерна інженерія. Магістерська програма - Комп’ютерна інженерія" / О.М. Березький, Л.О. Дубчак, Г.М. Мельник /Під ред. О.М. Березького – Тернопіль: ТНЕУ, 2018.– 41 с.

40. Нейфах А.Э. Сверточные коды для передачи дискретной информации. – М.: Наука, 1979. – 222 с.

41. Обобщенные каскадные помехоустойчивые конструкции на базе сверточных кодов. В.В. Зяблов, С.А. Шавгулидзе. - М.: Москва, 1991. – 207 с.

42. Романов В.О. Безпровідна сенсорна мережа для прецизійного землеробства та екологічного моніторингу / В.О.Романов, О.В. Палагін, І.Б. Галелюка, О.В. Вороненко // Комп'ютерні засоби, мережі та системи. – 2014,. – №13. – С.53-62.

43. Сетевое кодирование / Габидулин Э.М., Пилипчук Н.И., Колыбельников А.И. [та др.] // Труды МФТИ. – 2009. – Том 1, № 2 – С.3-28.

44. Скляр Б. Цифровая связь. Теоретические основы и практическое применение, 2-е издание.: Пер. с англ. – М.: Издательський дом "Вильямс», 2003. – 1104 с.

45. Столлингс В. Беспроводные линии связи и сети.: Пер. с англ. – М.: Издательський дом «Вильямс», 2003. – 640 с.

46. Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5- е изд. – СПб.: Питер, 2012. – 960 с.

47. Цаволик Т. Г., Яцків В. В. Метод формування корегувальних кодів у системі залишкових класів. Науковий вісник НЛТУ України. – 2017. – Вип. 27(3). – С. 191–194.

48. Цаволык Т.Г. Метод исправления ошибок на основе модулярных корректирующих кодов / Т.Г.Цаволык, В.В.Яцкив // Физика, математика, информатика. Вестник Брестского государственного технического университета, Брест. – 2015. – № 5 (850). – С. 36 – 38.

49. Цыбизов, А. А. Оценка эффективности сетей связи // Вестник Рязанского государственного радиотехнического университета. – 2009. – Вып. 3(29). – С. 19–24.

50. Чекмарев Ю. В. Вычислительные системы, сети и телекоммуникации. Издание второе, исправленное и дополненное. – М.: ДМК Пресс, 2009. – 184 с.

51. Червяков Н. И., Сахнюк П. А., Шапошников А. В., Ряднов С. А. Модулярные параллельные вычислительные структуры нейро- процессорных систем / Под. ред. Н.И. Червякова. – М.: ФИЗМАТЛИТ, 2003. – 288 с.

52. Шевчук Б.М., Задірака В.К., Гнатів Л.О., Фраєр С.В. Технологія багатофункціональної обробки і передачі інформації в моніторингових мережах. – К.: Наук. думка, 2010. – 370 с.

53. Яцків В. В. Методи виконання модулярних операцій та їх реалізація на ПЛІС / В. В. Яцків // Вісник Хмельницького національного університету. Технічні науки . - 2014. - № 6. - С. 218-224.

54. Яцків В.В. Виявлення та виправлення багатократних помилок на основі модулярних коректуючих кодів. Інформаційні технології та комп’ютерна інженерія. – 2015. – Том 33, №2. – С.77-82.

55. Яцків В.В. Модифіковані коректуючі коди системи залишкових класів та їх застосування / В.В. Яцків // Інформаційні технології та комп’ютерна інженерія. – 2013 – №2. – С.39-45.