Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ПОЯСНЮВАЛЬНА ЗАПИСКА
дипломної роботи

магістра
(назва освітнього рівня)

галузь знань *17 Електроніка та телекомунікації*
(шифр і назва галузі знань)

спеціальність *172 Телекомунікації та радіотехніка*
(код і назва спеціальності)

освітній рівень *магістр*
(назва освітнього рівня)

кваліфікація *магістр з телекомунікацій та радіотехніки*
(код і назва кваліфікації)

На тему: «Імітаційне моделювання характеристик кодеків»

Виконавець: студент *6* **курсу, групи** 172м-19з-1

(підпис) Малолітній Владислав Сергійович (прізвище ім'я по-батькові)

Керівники	Прізвище, ініціали	Оцінка	Підпис
проекту	к.ф.-м.н., проф. Гусєв О.Ю.		
розділів:			
спеціальний	к.ф.-м.н., проф. Гусєв О.Ю.		
економічний	к.е.н., доц. Романюк Н.М.		
Рецензент			
Нормоконтроль	к.ф.-м.н., проф. Гусєв О.Ю.		

Дніпро
2020

Міністерство освіти і науки України
Національний технічний університет
«Дніпровська політехніка»

Інститут електроенергетики
Факультет інформаційних технологій
Кафедра безпеки інформації та телекомунікацій

ЗАТВЕРДЖЕНО:
завідувач кафедри
безпеки інформації та телекомунікацій
д.т.н., професор_____Корнієнко В.І.
«_____»_____20___року

ЗАВДАННЯ
на дипломну роботу магістра

спеціальність *172 Телекомунікації та радіотехніка*
(код і назва спеціальності)

студента 172м-19з-1 Малолітній Владислав Сергійович
(група) (прізвище ім'я по-батькові)

Тема дипломного проекту «Імітаційне моделювання характеристик кодеків»

Наказ ректора НТУ "ДП" від _____№_____

Розділ	Зміст	Термін виконання
Стан питання. Постановка задачі	Аналітичний огляд літератури по темі проекту	Вересень 2020
Спеціальна частина	Принцип роботи систем передачі даних на базі технології розширення спектру прямою послідовністю. Розробка моделей кодеків в середі MATLAB. Оформлення пояснювальної записки.	Жовтень 2020
Економічний розділ	Розрахунок капітальних витрат	Листопад 2020

Завдання видав _____ Гусєв О.Ю.
(підпис) (прізвище, ініціали)

Завдання прийняв до виконання _____ Малолітній В.С.
(підпис) (прізвище, ініціали)

Дата видачі завдання: 03 вересня 2020 р.

Строк подання дипломного проекту до ДЕК:

РЕФЕРАТ

Пояснювальна записка: с., рис., табл., додатків, джерел.

Об'єкт розробки: системи прийому-передачі даних.

Предмет розробки: імітаційне моделювання характеристик кодеків.

Мета дипломного проекту: дослідження характеристик кодеків шляхом імітаційного моделювання.

У першому розділі виконаний аналітичний огляд літературних джерел по темі дипломної роботи. Здійснено постановку задачі роботи.

У другому розділі розроблено структуру та імітаційну модель найбільш широко застосовуваних кодеків. Виконано модельний експеримент і проведено аналіз результатів.

У третьому розділі виконано розрахунок капітальних витрат на розробку імітаційних моделей кодеків.

СИСТЕМИ ПЕРЕДАЧІ ДАНИХ, МОДЕЛЬ СИСТЕМИ, ІМІТАЦІЙНЕ МОДЕЛЮВАННЯ, МОДУЛЯЦІЯ, МАНІПУЛЯЦІЯ, КОДЕК

РЕФЕРАТ

Пояснительная записка: с., рис., табл., прилож., источн..

Объект разработки: системы приема-передачи данных.

Предмет разработки: имитационное моделирование характеристик кодеков.

Цель дипломного проекта: исследование характеристик кодеков путем имитационного моделирования

В первой главе выполнен аналитический обзор литературных источников по теме дипломной работы. Осуществлена постановка задачи работы.

Во втором разделе разработана структура и имитационную модель наиболее широко применяемых кодеков. Выполнен модельный эксперимент и проведен анализ результатов.

В третьем разделе выполнен расчет капитальных затрат на разработку имитационных моделей кодеков.

СИСТЕМЫ ПЕРЕДАЧИ ДАННЫХ, МОДЕЛЬ СИСТЕМЫ, ИМИТАЦИОННОЕ МОДЕЛИРОВАНИЕ, МОДУЛЯЦИЯ, МАНИПУЛЯЦИЯ, КОДЕК

ABSTRACT

Explanatory note: p., fig., tables, app., sources.

Development object: data reception and transmission systems.

Subject of development: simulation of the characteristics of codecs.

The purpose of the thesis project: study of the characteristics of codecs by simulation

The first chapter contains an analytical review of literary sources on the topic of the thesis. The task of the work was formulated.

In the second section, the structure and simulation model of the most widely used codecs are developed. A model experiment was carried out and the results were analyzed.

In the third section, the calculation of capital costs for the development of simulation

models of codecs is performed.

DATA TRANSMISSION SYSTEMS, SYSTEM MODEL, SIMULATION, MODULATION, MANIPULATION, CODEC

СПИСОК УМОВНИХ СКОРОЧЕНЬ

BPSK - (Binary phase-shift keying) Двійкова фазова модуляція

FSK - (Frequency Shift Keying) Частотна маніпуляція

IC - Інтегральна схема

CRC - (Cyclic redundancy code) Циклічний надлишковий код

РЕБ - Радіоелектронна боротьба

SNR - (Signal to Noise Ratio) Відношення сигнал-шум

СПДІ - Системи передачі дискретної інформації

СРЗ - Системи радіозв'язку

КК - Коригувальні коди

БЧХ - Боуза-Чоудхурі-Хоквенгема код

ІС - інтегральні схеми

КС – канал зв'язку

ЗМІСТ

ВСТУП .....
1 СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ .....
1.1. Коригувальні коди, що застосовуються в телекомунікаційних системах. Класифікація коригувальних кодів. .....
1.2 Аналітичний огляд основних коригувальних кодів .....
1.2.1 Коди Хеммінга, БЧХ (Боуза-Чоудхурі-Хоквенгема), Ріда Соломона .....
1.2.2 Циклічні надлишкові коди CRC .....
1.2.3 Згорткові коди .....
1.3 Постановка задачі .....
1.4 Висновки .....
2 СПЕЦІАЛЬНА ЧАСТИНА .....
2.1 Моделювання кодека Хеммінга і дослідження його характеристик .....
2.2 Моделювання кодека БЧХ та дослідження його характеристик .....
2.3 Моделювання кодека Ріда-Соломона і дослідження його характеристик .....
2.4 Моделювання CRC-кодека і дослідження його характеристик……
2.5 Моделювання кодека на базі згорткового кодування і дослідження його характеристик .....
2.6 Висновки .....
3 ЕКОНОМІЧНА ЧАСТИНА .....
3.1 Розрахунок капітальніх витрат на розробку імітаційних моделей

кодеків.....
3.1.1 Визначення трудомісткості розробки моделі .....
3.1.2 Розрахунок витрат на розробку моделі .....
3.1.3 Розрахунок капітальніх витрат .....
3.2 Висновки .....
ВИСНОВКИ .....
ПЕРЕЛІК ПОСИЛАНЬ .....
Додаток А .....
Додаток Б .....
Додаток В .....

ВСТУП

Кодек забезпечує виявлення і виправлення помилок на приймальній стороні системи зв’язку за рахунок інформаційної надмірності, внесеної за певними правилами в передане повідомлення. Завданням кодує пристрої є перетворення переданого знака (фрагмента) повідомлення в кодову комбінацію.

Декодер на приймальній стороні каналу зв'язку (КС) здійснює зворотне перетворення кодових комбінацій в знаки (фрагменти) повідомлення. При наявності перешкод в каналі зв'язку існує ймовірність помилкового декодування знака (фрагмента). Ця ймовірність залежить від ймовірності помилкового прийому елементів повідомлення і від надмірності коду.

При наявності в кодової комбінації певної надмірності на приймальній стороні каналу зв'язку можливе виявлення помилково прийнятих знаків (фрагментів) повідомлення, а якщо надмірність коду досить велика, то можливо не тільки виявлення помилок, але і відновлення знаків (фрагментів) повідомлення в первісному вигляді.

За частости виявлених помилок в знаках (фрагментах) повідомлення можна здійснювати оцінку якості прийому окремих індивідуальних пакетів і всього повідомлення в цілому.

На вході модулятора і декодирующего пристрої мають місце бінарні послідовності імпульсів, тривалості яких кратні певній величині, обумовленої інформаційної швидкістю передачі повідомлення, вимірюваної в біт / с.

Для розробки і порівняння ефективності кодеків необхідно вміти моделювати одновимірний канал зв'язку з дискретними входом і дискретним виходом.

РОЗДІЛ 1. СТАН ПИТАННЯ. ПОСТАНОВКА ЗАДАЧІ

1.1 Коригувальні коди, що застосовуються в телекомунікаційних системах. Класифікація коригувальних кодів

В теорії сучасних телекомунікаційних систем значна увага приділяється вивченню методів кодування інформації. У техніці електричного зв'язку широко використовують результати теорії кодування.

Кодування - операція ототожнення символів або груп символів одного коду з символами або групами символів іншого коду. Необхідність кодування виникає, перш за все, з потреби пристосувати форму повідомлення до даного каналу зв'язку або до будь-якого іншого пристрою, призначеного для перетворення або зберігання інформації.

Типова структурна схема системи передачі дискретної інформації (СПДІ) приведена на рис. 1.1. Джерело виробляє повідомлення, які необхідно передавати по каналу СПДІ. Це можуть бути послідовності дискретних повідомлень (дані, телеграфні повідомлення і т. д.) або безперервні повідомлення (мова, телебачення, результати телевимірювань та ін.), перетворені в цифрову форму.

Реальні повідомлення містять надмірність і для узгодження джерела з каналом передачі інформації використовують кодер джерела. Спільно з декодером вони утворюють кодек джерела. Методи кодування джерел не є предметом дипломної роботи. Основне завдання будь-якої СДПІ - передача

інформації з заданими вірністю і швидкістю передачі. Ці вимоги знаходяться в протиріччі, причому, підвищення швидкості передачі інформації в реальних СПДІ призводить до зниження завадостійкості і вірності передачі.

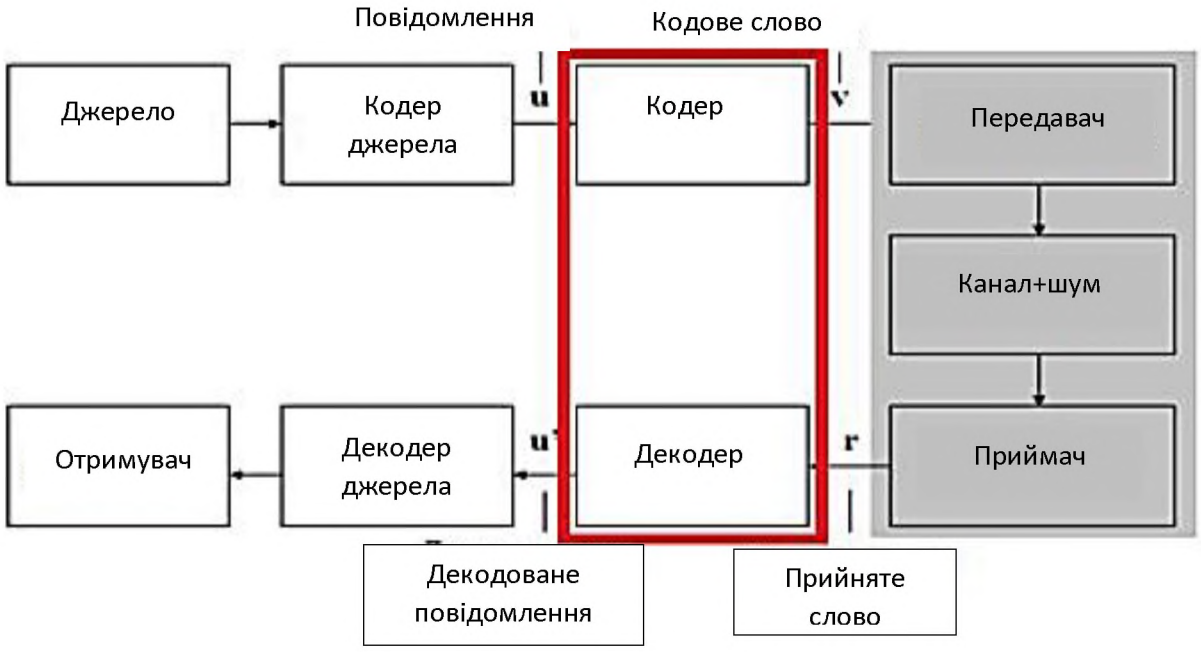

Рисунок 1.1 – Структурна схема системи передачі дискретної інформації

Згідно відомих теорем К. Шеннона, в принципі можливо як завгодно велике підвищення вірності передачі інформації, якщо швидкість передачі по каналу R_{Kan} не перевищує пропускної здатності каналу $C_к$. Досягається це застосуванням досить довгих коригувальних кодів (КК).

Коригувальні коди - це коди, що дозволяють виявляти або виправляти помилки, що виникають при передачі повідомлень по каналах зв'язку.

З цією метою в структуру КК вводиться надмірність.

Кодек КК (кодер і декодер каналу) наведені на рис.1.1. В реальних умовах довжина коду обмежена допустимою складністю пристроїв кодування і, перш за все, декодування, тому ефект від застосування коригувальних кодів залежить від параметрів коду і обмежень на реалізацію кодека каналу.

Сучасна теорія пропонує широкий набір коригувальних кодів, різних за структурою, принципами побудови і коректуючою здібністю. Нижче розглянуті класи кодів, для яких розроблені досить прості та ефективні алгоритми кодування/декодування, і які є найбільш перспективними для використання в каналах телекомунікаційних систем.

Класифікація коригувальних кодів

У теорії і техніці завадостійкого кодування відома безліч коригувальних кодів, які можуть бути класифіковані за різними ознаками.

Класифікація кодів наведена на рис. 1.2.

За способом формування КК поділяються на блокові і безперервні. Формування блокових кодів передбачає розбиття переданих цифрових послідовностей на окремі блоки, які подаються на вхід кодера. Кожному такому блоку на виході кодера відповідає блок кодових символів, робота кодера визначається правилом, або алгоритмом кодування. Формування безперервних кодів здійснюється безперервно в часі, без поділу на блоки, що і визначає найменування цього класу кодів. Блокові коди історично були запропоновані і вивчені раніше, на зорі розвитку теорії кодування.

У класі безперервних кодів слід зазначити згорткові коди, які за характеристиками перевершують блокові коди, і, з цієї причини, знаходять широке застосування в телекомунікаційних системах. Багато кодів носять імена вчених, які їх запропонували і досліджували. Таким прикладом є безперервний код Фінка-Хагельбаргера, запропонований радянським вченим Л.М. Фінком і німецьким фахівцем Р. Хагельбаргером. Тривалий час цей код служив в літературі показовим прикладом безперервного коду з простим алгоритмом кодування/декодування, але після відкриття згорткових кодів поступився їм місцем.

Для опису процедур кодування/декодування як блокових, так і згорткових кодів використовують адекватний математичний апарат. Для

опису лінійних кодів використовується добре розроблений апарат лінійної алгебри. Формування нелінійних кодів проводиться із застосуванням нелінійних процедур. Такий підхід дозволяє в деяких випадках отримати нелінійні коди з рядом спеціальних властивостей. У теорії і техніці кодування важливою є проблема складності реалізації процедур кодування/декодування і, особливо, процедур декодування. Тому деякі класи кодів (коди Хеммінга, циклічні коди Боуза-Чоудхурі-Хоквінгема, Ріда-Соломона, Файра та ін.) були розроблені спільно з алгоритмами декодування, пов'язаними зі структурними властивостями цих кодів. І, навпаки, розробка нових алгоритмів декодування згорткових кодів (алгоритм А. Вітербу, послідовне декодування, порогове декодування) ініціювала пошуки відповідних згорткових кодів.

Відмінні переваги коригувальних кодів (як блокових, так і згорткових) спонукали пошуки нових підходів до реалізації шляхів підвищення завадостійкості і ефективності телекомунікаційних систем. На рис. 1.2 відзначені, відповідно, нові методи кодування: сигнально-кодові конструкції, турбокоди, просторово-часові коди і т.п.

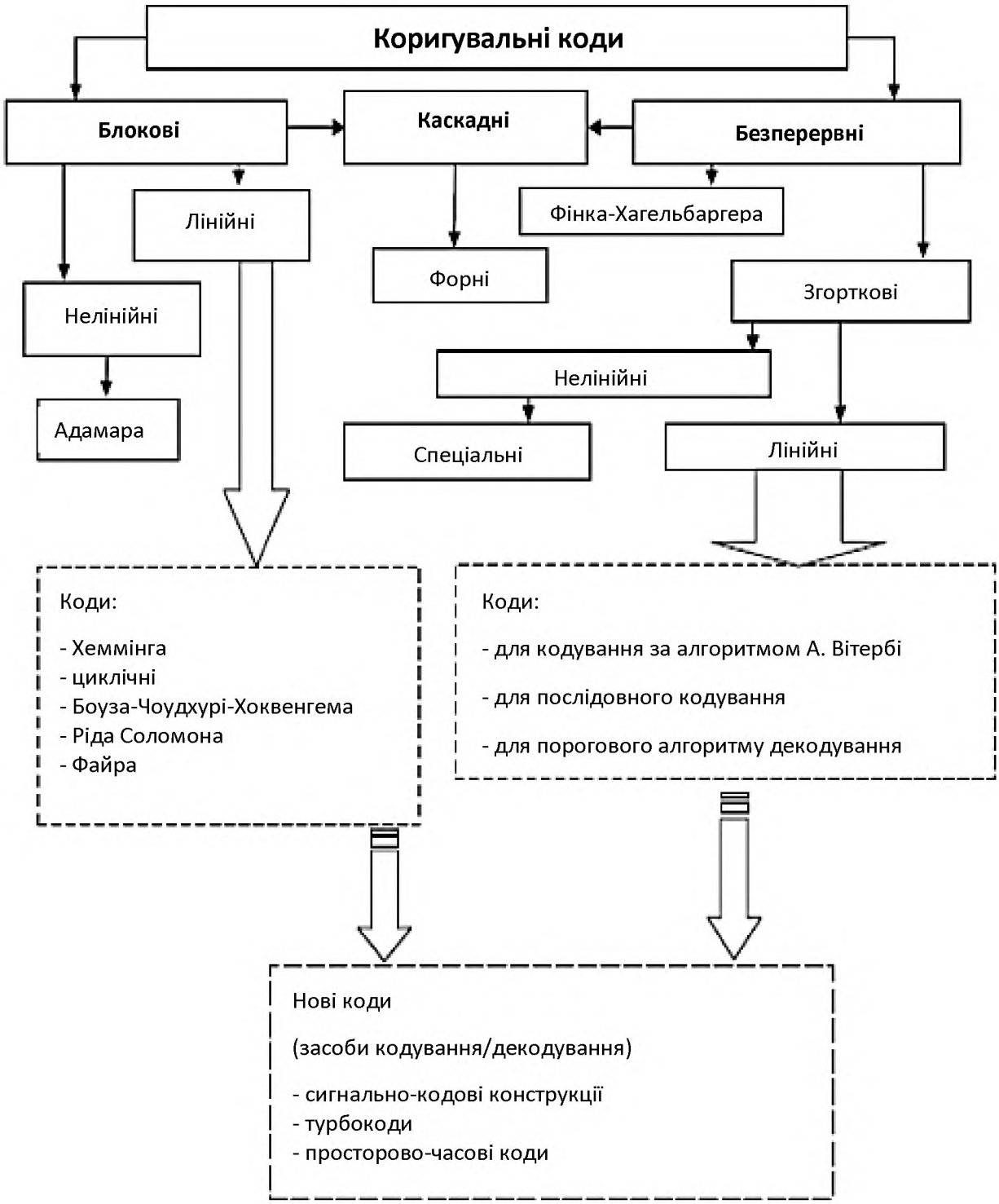

Рисунок 1.2 – Класифікація коригувальних кодів

1.2 Аналітичний огляд основних коригувальних кодів

1.2.1 Коди Хеммінга, БЧХ (Боуза-Чоудхурі-Хоквенгема), Ріда Соломона

Код Хеммінга

Найбільш відомим лінійним кодом є блоковий код Хеммінга. В даному пункті буде розглянуто (7,4) - код Хеммінга. Лінійний блоковий код, для якого виконується нерівність Хеммінга, є кодом Хеммінга, тобто код (7,4) не єдиний код, який є цим кодом.

Введемо деякі позначення, які складають основу завадостійкого кодування:

k - кількість інформаційних біт

n - кількість біт на виході кодера

r - кількість перевірочних (надлишкових) біт

Замість k біт інформаційного вектора в канал передається n біт кодового вектора. У цьому випадку говорять про надмірне кодування зі швидкістю: $R = n/k$.

Чим нижче швидкість, тим більше надмірність коду, і тим більшими можливостями для захисту від помилок він володіє. Однак, слід враховувати, що зі збільшенням надмірності витрати на передачу інформації також зростають.

У кодуванні і декодуванні беруть участь генераторна і перевірочна матриці, структура яких наступна:

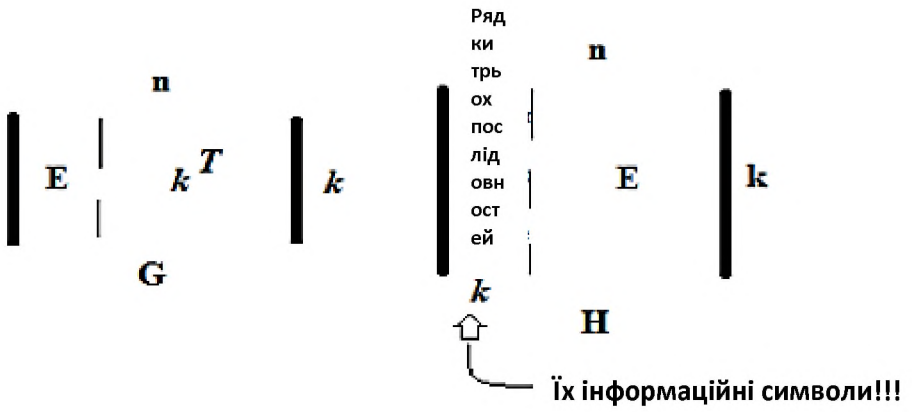

Рисунок 1.3 – Генераторна і перевірочна матриці

Також, код Хеммінга має мінімальну кодову відстань в просторі, рівну трьом, з цього випливає, що даний код вміє виявляти будь-які дворазові помилки і виправляти будь-які одноразові.

Розглянемо код Хеммінга (7,4) в теорії.

Для цього коду генераторна та перевірочна матриці вже заготовлені і мають такий вигляд:

$$G_{4x7} = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} H_{3x7} = \begin{pmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{pmatrix}$$

Рисунок 1.4 – Структура генераторної і перевірочної матриці для коду (7,4)

Кодування здійснюється наступним чином:

Кодове слово n і інформаційне слово k пов'язані співвідношенням:

$$N = v = k*G$$

де G – породжуюча матриця, структура якої була описана вище.

Наприклад, інформаційний вектор k = (1010) відобразиться в кодовий вектор наступним чином:

$$v = (1010) \times \begin{pmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{pmatrix} = (0011010)$$

Рисунок 1.5 – Кодування коду (7,4)

Легко помітити, що останні чотири розряду кодового вектора збігаються з інформаційним вектором. Ця властивість називається систематичністю коду. Тут перші три біта послідовності є перевірочними (надлишковими).

Декодування здійснюється на отриманні синдрому і проводиться це все наступним чином:

Система перевірочних рівнянь виглядає:

$$s = n = v*H^T$$

Вектор s прийнято називати синдромом. Таким чином, помилка буде виявлена, якщо хоча б одна з компонент s не дорівнює нулю.

При передачі інформаційного слова a = (1010) по каналу без шуму вихідний вектор був n = (0011010). Можемо переконатися, що в цьому випадку синдром дорівнює 0.

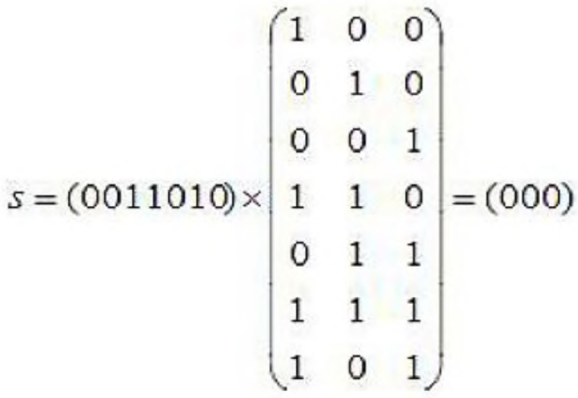

Рисунок 1.6 – Отримання синдрому для коду (7,4)

Якщо, наприклад, в кодовому слові сталася одиночна помилка на четвертій позиції (r = (0010010)), то синдромом є четвертий рядок транспонованої перевірочної матриці

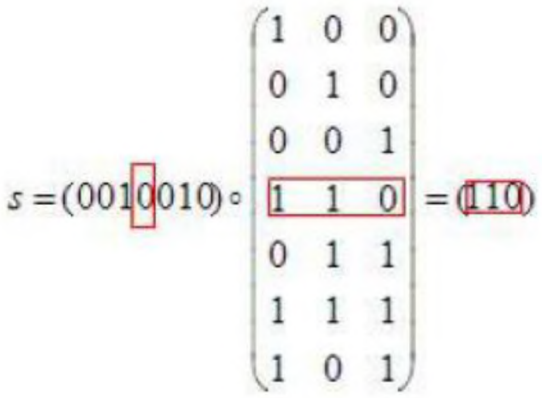

Рисунок 1.7 – Знаходження помилкового біта для коду (7,4)

Перебравши всі можливі позиції одиночної помилки, отримаємо повну таблицю синдромів одноразових помилок - таблицю відповідностей номера помилкового розряду, отримуваному при цьому синдрому. Таким чином проводиться декодування. Якщо помилки знайдені, то вони відповідно виправляються.

Код БЧХ (Боуза-Чоудхурі-Хоквенгема)

Код БЧХ є циклічним кодом. З циклічним кодом Хеммінга у них багато спільного, а саме алгоритм кодування, який відрізняється тільки

знаходженням генераторного полінома, а процес декодування повністю схожий. Необходімо почати з невеликого введення в код БЧХ. Многочлен *g (x)* ступеня називається примітивним, якщо $x^j + 1$ ділиться на *g(x)* без залишку для $j = 2^k - 1$ і не ділиться ні для якого меншого значення (де *k* - кількість інформаційних біт). Наприклад, многочлен $1+x^2+x^3$ примітивний: він ділить x^7+1, але не ділить x^j+1 при $j < 7$.

Кодуючий многочлен *g(x)* для БЧХ-коду, довжина кодових слів якого *n*, будується так. Знаходиться примітивний многочлен мінімального ступеня такий, що $n \le 2^q - 1$ або $q \ge \log_2(n+1)$. Нехай α - корінь цього многочлена, тоді розглянемо кодуючий многочлен $g(x)$ = НОК($m_1(x)$, …, $m_{d-1}(x)$), где $m_1(x),\ldots,m_{d-1}(x)$ - многочлени мінімального ступеня, що мають корінням відповідно $\alpha^2,\ldots, \alpha^{d-1}$.

Побудований кодуючий многочлен виробляє код з мінімальною відстанню між кодовими словами, не меншою ***d***, і довжиною кодових слів ***n***.

Кодування:

Наприклад, потрібно побудувати БЧХ-код з довжиною кодових слів n = 15 і мінімальною відстанню між кодовими словами d = 5. Ступінь примітивного многочлена дорівнює $q = \log_2(n+1) = 4$ і сам він дорівнює $1+x^3+x^4$ (примітивний многочлен 4-го ступеня для коду (7,15)). Нехай α - його корінь, тоді 2 і 4 - також його коріння. Мінімальним многочленом для α^3 буде $1+x+x^2+x^3+x^4$. Отже $g(x)$ = НОК($1+x^3+x^4$, $1+x+x^2+x^3+x^4$) = $=(1+x^3+x^4)(1+x+x^2+x^3+x^4) = 1+x+x^2+x^4+x^8$.

Ступінь отриманого многочлена дорівнює 8, побудований БЧХ-код буде (7,15) кодом. Слово 1000100 або $a(x) = x^4+1$ буде закодовано кодовим словом $a(x)g(x) = x^{12}+x^6+x^5+x^2+x+1$ або 111001100000100.

Декодування:

Декодування проводиться шляхом ділення закодованої послідовності на генераторний поліном, який використовувався при кодуванні. Отримана послідовність при розподілі і буде декодованою послідовністю. У кращому випадку, залишку при діленні не буде, це означає, що помилок не виявлено.

Якщо ж залишок є, називається він не інакше як синдром, в такому випадку помилки присутні в закодованій послідовності. В цьому випадку поступають так.

Закодовану послідовність складають з вектором помилок по модулю два і виправляють помилковий біт. Вектор помилок формується за допомогою спеціальних схем, які, аналізуючи закодовану послідовність, формують даний вектор.

Ще одним способом виправлення помилок є наступний метод. На підставі отриманого генераторного полінома будується схема і на кожному такті її роботи визначається синдром. В даному випадку необхідно отримати два синдрому, один з них це залишок, від поділу отриманий при декодуванні, а другий - це комбінація 100. Далі проводиться віднімання номера такту комбінації 100 і номера такту залишку. Отримана різниця і є номером біта в закодованої комбінації.

Код Рида-Соломона

Кодувальник Ріда-Соломона бере блок цифрових даних і додає додаткові «надлишкові» біти. Помилки відбуваються при передачі по каналах зв'язку або з різних причин при запам'ятовуванні (наприклад, через шум або наведення, подряпини на CD і т.д.).

Декодер Ріда-Соломона обробляє кожен блок, намагається виправити помилки і відновити вихідні дані. Число і типи помилок, які можуть бути виправлені, залежать від характеристик коду Ріда-Соломона.

Властивості кодів Ріда-Соломона

Коди Ріда-Соломона є субнабором кодів БЧХ і являють собою лінійні блокові коди. Код Ріда-Соломона специфікуються як RS (n, k) s - бітних символів.

Це означає, що кодувальник сприймає k інформаційних символів по s бітів кожен і додає символи парності для формування ***n*** символьного кодового слова.

Декодер Ріда-Соломона може коригувати до ***t*** символів, які містять помилки в кодовому слові, де $2t = n-k$.

Діаграма, представлена нижче, показує типове кодове слово Ріда-Соломона:

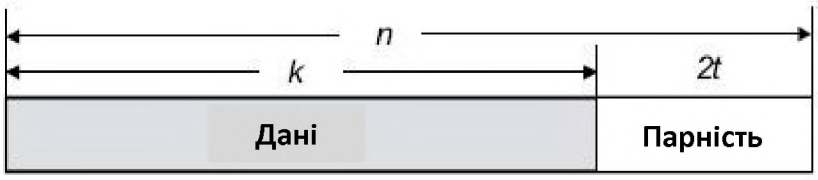

Рисунок 1.8 – Кодове слово для коду Р-С

Популярним кодом Ріда-Соломона є RS (255, 223) з 8-бітними символами.

Кожне кодове слово містить 255 байт, з яких 223 є інформаційними і 32 байтами парності. Для цього коду ***n = 255, k = 223, s = 8, 2t = 32, t = 16.***

Декодер може виправити будь-які 16 символів з помилками в кодовому слові: тобто помилки можуть бути виправлені, якщо число спотворених байт не перевищує 16.

При розмірі символу s максимальна довжина кодового слова ***n*** для коду Ріда-Соломона дорівнює $n = 2^s - 1$.

Наприклад, максимальна довжина коду з 8-бітними символами (s = 8) дорівнює 255 байтам.

Коди Ріда-Соломона можуть бути в принципі вкорочені шляхом обнулення деякого числа інформаційних символів на вході кодувальника (передавати їх в цьому випадку не потрібно). При передачі даних декодера ці нулі знову вводяться в масив.

Код (255, 223), описаний вище, може бути укорочений до (200, 168). Кодувальник буде працювати з блоком даних 168 байт, додасть 55 нульових байт, сформує кодове слово (255, 223) і передасть тільки 168 інформаційних байт і 32 байта парності.

Обсяг обчислювальної потужності, необхідної для кодування і декодування кодів Ріда-Соломона, залежить від числа символів парності. Велике значення ***t*** означає, що більша кількість помилок може бути виправлена, але це зажадає більшої обчислювальної потужності в порівнянні з варіантом при меншому ***t***.

Помилки в символах

Одна помилка в символі відбувається, коли 1 біт символу виявляється невірним або коли всі біти невірні.

Код RS (255,223) може виправити до 16 помилок в символах. У гіршому випадку, можуть мати місце 16 бітових помилок в різних символах (байтах). У кращому випадку, коригуються 16 повністю невірних байт, при цьому виправляється 16 * 8 = 128 бітових помилок.

Коди Ріда-Соломона особливо добре підходять для коригування кластерів помилок (коли невірними виявляються великі групи біт кодового слова, які йдуть поспіль).

Декодування

Алгебраїчні процедури декодування Ріда-Соломона можуть виправляти помилки і втрати. Втратою вважається випадок, коли положення невірного символу відомо. Декодер може виправити до t помилок або до 2t втрат. Дані

про втрати (стирання) можуть бути отримані від демодулятора цифрової комунікаційної системи, тобто демодулятор позначає отримані символи, які ймовірно містять помилки.

Коли кодове слово декодується, можливі три варіанти.

1 Якщо $2s + r < 2t$ (s помилок, r втрат), тоді вихідне передане кодове слово завжди буде відновлено. В іншому випадку:

2 Декодер детектує ситуацію, коли він не може відновити вихідне кодове слово. Або ж:

3 Декодер некоректно декодує і невірно відновить кодове слово без якої-небудь вказівки на цей факт.

Імовірність кожного з цих варіантів залежить від типу використовуваного коду Ріда-Соломона, а також від числа і розподілу помилок.

Архітектура кодування і декодування кодів Ріда-Соломона

Кодування і декодування Ріда-Соломона може бути виконано апаратно або програмно.

Арифметика кінцевого поля Галуа

Коди Ріда-Соломона базуються на спеціальному розділі математики - полях Галуа (GF) або кінцевих полях. Арифметичні дії (+, -, x, / і т.д.) над елементами кінцевого поля дають результат, який також є елементом цього поля. Кодувальник або декодер Ріда-Соломона повинні вміти виконувати ці арифметичні операції. Ці операції для своєї реалізації вимагають спеціального обладнання або спеціалізованого програмного забезпечення.

Утворюючий поліном

Кодове слово Ріда-Соломона формується із залученням спеціального полінома.

Все коректні кодові слова повинні ділитися без залишку на ці утворюючі поліноми.

Загальна форма утворюючого полінома має вигляд

$$g(x) = (x - a^{i})(x - a^{i+1})\dots(x - a^{i+1+2t}),$$

а кодове слово формується за допомогою операції

$$c(x) = g(x) * i(x),$$

де *g(x)* є утворюючим поліномом, *i(x)* являє собою інформаційний блок, *c(x)* - кодове слово, зване простим елементом поля.

Архітектура кодувальника

2t символів парності в кодовому слові Ріда-Соломона визначаються з наступного співвідношення:

$$p(x) = i(x) * x^{n-k} \bmod g(x)$$

Нижче показана схема реалізації кодувальника для версії RS (255,249):

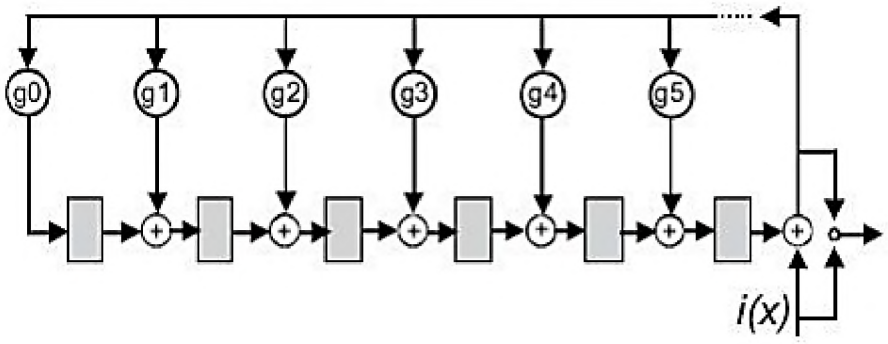

Рисунок 1.9 – Схема кодувальника Р-С

Архітектура декодера

Кожен з 6 регістрів містить в собі символ (8 біт). Арифметичні оператори виконують додавання множення на символ як на елемент кінцевого поля.

Загальна схема декодування кодів Ріда-Соломона показана нижче на рис. 1.10.

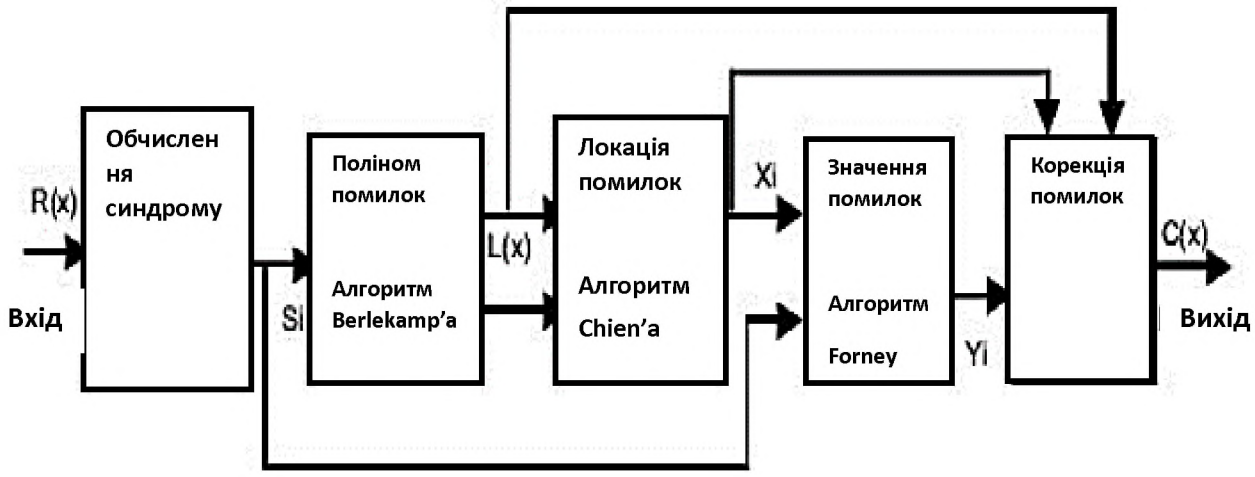

Рисунок 1.10 – Схема роботы з кодами Ріда-Соломона

Позначення:

r(x) - отримане кодове слово

Si - синдроми

L(x) - поліном локації помилок

Xi - положення помилок

Yi - значення помилок

c(x) - відновлене кодове слово

v - число помилок

Отримане кодове слово r(x) являє собою вихідне (передане) кодове слово c(x) плюс помилки:

$$r(x) = c(x) + e(x)$$

Декодер Ріда-Соломона намагається визначити позицію і значення помилки для ***t*** помилок (або ***2t*** втрат) і виправити помилки і втрати.

Обчислення синдрому

Обчислення синдрому схоже на обчислення парності. Кодове слово Ріда-Соломона має ***2t*** синдромів, це залежить тільки від помилок (а не переданих кодових слів). Синдроми можуть бути обчислені шляхом підстановки ***2t*** коренів утворюючого полінома g(x) в r(x).

Знаходження позицій символьних помилок

Це робиться шляхом вирішення системи рівнянь з ***t*** невідомими. Існує кілька швидких алгоритмів для вирішення цього завдання. Ці алгоритми використовують особливості структури матриці кодів Ріда-Соломона і сильно скорочують необхідну обчислювальну потужність. Робиться це в два етапи.

1 Визначення полінома локації помилок.

Це може бути зроблено за допомогою алгоритму Berlekamp-Massey або алгоритму Евкліда. Алгоритм Евкліда використовується частіше на практиці, так як його легше реалізувати, проте алгоритм Berlekamp-Massey дозволяє отримати більш ефективну реалізацію обладнання та програм.

2 Знаходження коренів цього полінома. Це робиться з залученням алгоритму пошуку Chien.

Знаходження значень символьних помилок

Тут також потрібно вирішити систему рівнянь з t невідомими. Для вирішення використовується швидкий алгоритм Forney.

Реалізація кодувальника і декодера Ріда-Соломона. Апаратна реалізація

Існує кілька комерційних апаратних реалізацій. Є багато розроблених інтегральних схем, призначених для кодування і декодування кодів Ріда-Соломона. Ці ІС допускають певний рівень програмування (наприклад, RS (255, k), де t може приймати значення від 1 до 16).

Програмна реалізація

До недавнього часу програмні реалізації в «реальному часі» вимагали занадто великої обчислювальної потужності практично для всіх кодів Ріда-Соломона.

Головними труднощами в програмній реалізації кодів Ріда-Соломона було те, що процесори загального призначення не підтримують арифметичні операції для поля Галуа.

Однак оптимальне складання програм в поєднанні із збільшеною обчислювальною потужністю дозволяють отримати цілком прийнятні результати для відносно високих швидкостей передачі даних.

1.2.2 Циклічні надлишкові коди CRC

Циклічний надлишковий код (Cyclic redundancy code, CRC) - алгоритм обчислення контрольної суми, призначений для перевірки цілісності переданих даних. Алгоритм CRC виявляє все поодинокі помилки, подвійні помилки і помилки в непарному числі біт. Поняття циклічних кодів досить широке, однак на практиці його зазвичай використовують для позначення тільки одного різновиду, що використовує циклічний контроль (перевірку) надмірності.

Це дуже потужний і широко використовуваний метод виявлення помилок передачі інформації. Він забезпечує виявлення помилок з високою

ймовірністю. Крім того, цей метод має ряд інших корисних моментів, які можуть знайти своє втілення в практичних завданнях.

Головна особливість значення CRC полягає в тому, що він однозначно ідентифікує вихідну бітову послідовність і тому використовується в різних протоколах зв'язку, а також для перевірки цілісності блоків даних, переданих різними пристроями. Завдяки відносній простоті алгоритм обчислення CRC часто реалізується на апаратному рівні.

При передачі пакетів з мережевого каналу можуть виникнути спотворення вихідної інформації внаслідок різних зовнішніх впливів: електричних наводок, поганих погодних умов і багатьох інших. Сутність методики в тому, що при хороших характеристиках контрольної суми в переважній кількості випадків помилка в повідомленні призведе до зміни його контрольної суми. Якщо вихідна і обчислена суми не рівні між собою, приймається рішення про недостовірність прийнятих даних, і можна запросити повторну передачу пакета.

Основна ідея алгоритму CRC полягає в поданні повідомлення у вигляді величезного двійкового числа, розподілі його на інше фіксоване двійкове число і використання залишку цього поділу в якості контрольної суми [1]. Отримавши повідомлення, приймач може виконати аналогічну дію і порівняти отриманий залишок з «контрольною сумою».

На рисунках 1.11-1.12 зображено графічне представлення кодування і декодування CRC-коду [2]:

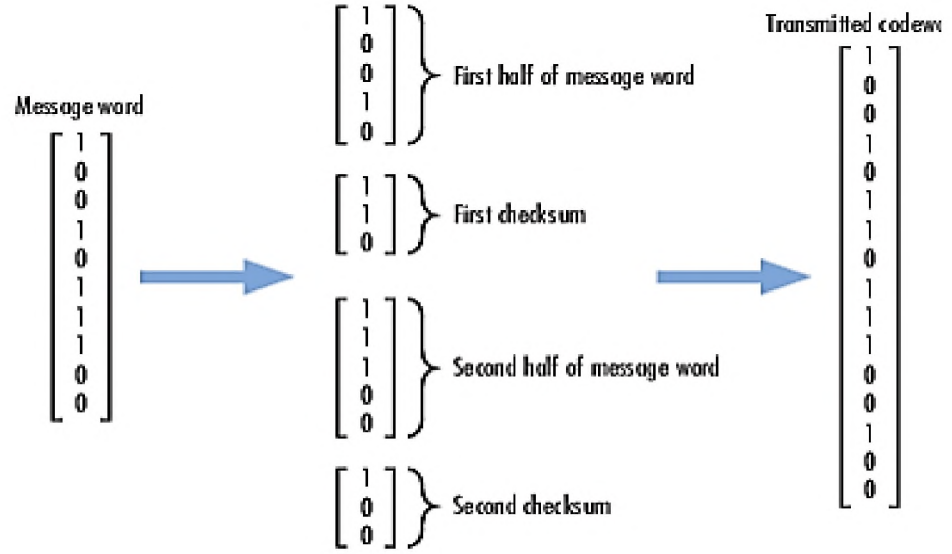

Рисунок 1.11 – Принцип роботи кодера CRC

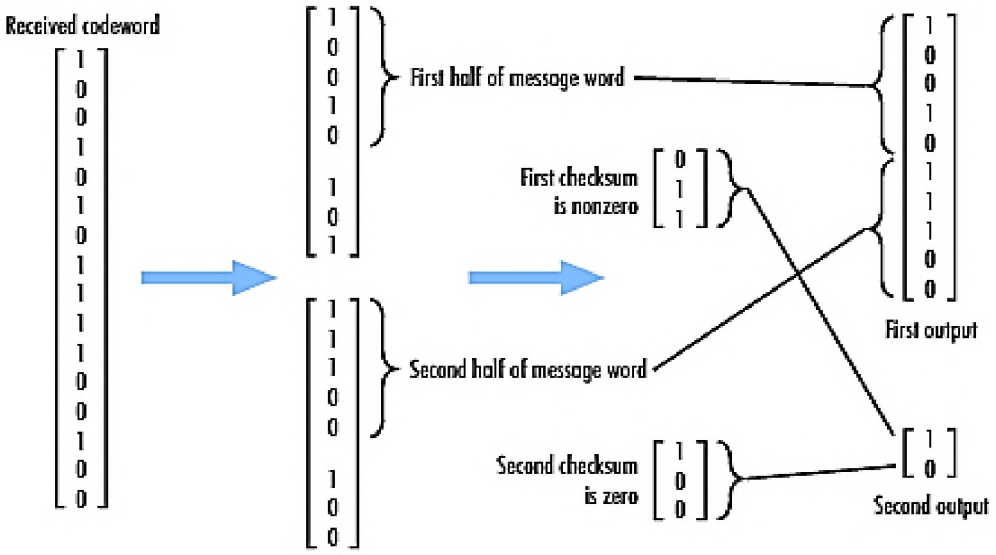

Рисунок 1.12 – Принцип роботи декодера CRC

Ступенем CRC-полінома W називають позицію самого старшого одиничного біта. Наприклад, ступінь полінома 10011_2 дорівнює 4.

Для обчислення CRC використовують поліноміальну арифметику. Замість подання дільника, діленого (повідомлення), результату ділення і залишку у вигляді позитивних цілих чисел, можна представити їх у вигляді

поліномів з двійковими коефіцієнтами або у вигляді рядка біт, кожен з яких є коефіцієнтом полінома.

Наприклад, десяткове число 23 в 16-річній і 2-їчній системах матиме вид $23_{10} = 17_{16} = 10111_2$, що збігається з поліномом:

$$1 \cdot x^4 + 0 \cdot x^3 + 1 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0$$

або спрощено:

$$x^4 + x^2 + x^1 + 1.$$

І повідомлення, і дільник можуть бути представлені у вигляді поліномів, з якими можна виконувати будь-які арифметичні дії без переносів.

Як правило, контрольна сума додається до вихідного повідомлення і отримане розширене повідомлення передається через канал зв'язку.

На іншому кінці каналу приймач може зробити одну з можливих дій (обидва варіанти абсолютно рівноправні):

1 Виділити текст отриманого повідомлення, обчислити для нього контрольну суму і порівняти її з переданою.

2 Обчислити контрольну суму для всього переданого повідомлення і подивитися, чи вийде в результаті нульовий залишок.

Оскільки вихідне повідомлення може бути дуже великим (до декількох Мбайт) і так само через те, що для отримання CRC використовується CRC-арифметика, використовувати звичайну комп'ютерну операцію ділення не можна.

Найпопулярніший і рекомендований IEEE поліном для CRC-32 використовується в Ethernet, FDDI; також цей многочлен є генератором коду Хеммінга.

1.2.3 Згорткові коди

Сучасна теорія кодів досить розвинена і містить детальну класифікацію. Всі вживані коди можна розбити на дві великі групи: блокові, в яких кодування та декодування проводиться в межах певної ділянки кодової послідовності-блоку, і деревовидні, в яких обробка символів проводиться безперервно, без поділу на блоки. Частина кодів відноситься до розряду лінійних, в яких кодові послідовності представлені як елементи лінійного векторного простору. Можна застосувати також розбиття на коди, що виправляють незалежні випадкові помилки, і коди, що виправляють пакетні помилки.

На відміну від блокових, згорткові коди мають наступні переваги:

- згорткові коди дозволяють виробляти кодування і декодування потоків даних безперервно в часі;

- згорткові коди не потребують блокової синхронізації;

- застосування згорткових кодів дозволяє досягти дуже високої надійності інформації, що передається.

Згорткові коди використовуються при низькому відношенні сигнал-шум, коли виправляючої здатності блокових кодів при розумній довжині блоку виявляється недостатньо.

Згорткове кодування найзручніше описувати, характеризуючи дію відповідного кодуючого пристрою. Згортковий кодер являє собою пристрій, що сприймає за кожен такт роботи в загальному випадку $к$ вхідних інформаційних символів і видає на вихід за той же такт n вихідних символів, що підлягають передачі по каналу зв'язку. Параметром згорткового коду, що характеризує його стійкість перед перешкодами, є мінімальна вільна відстань - d_c, яка визначається як мінімальна відстань по Хеммінгу між послідовностями згортаючого коду по довжині кодових обмежень по виходу.

Кодове обмеження по виходу - це число символів на виході кодера, в формуванні яких бере участь один вхідний біт.

Ефективність згорткового кода визначається в основному тим, яким чином пов'язані суматори з комірками регістра зсуву.

Відношення $R = k/n$ називають відносною швидкістю коду. Вихідні символи, створювані кодером на даному такті, залежать від k інформаційних символів, що надійшли на цьому і попередньому тактах. Таким чином, вихідні символи згорткового кодера однозначно визначаються його вхідним сигналом і станом, що залежить *від m-k* попередніх інформаційних символів.

Основними елементами згорткового кодера є: регістр зсуву, суматори за модулем 2 і комутатор.

Регістр зсуву є динамічним запам'ятовуючим пристроєм (рисунок 1.13), в якому зберігаються двійкові символи 0 або 1. Число тригерних комірок m в регістрі зсуву і визначає пам'ять коду. У момент надходження на вхід регістра нового інформаційного символу символ, що зберігається в крайньому правому розряді, виводиться з реєстру і скидається. Кожен символ з решти, що зберігаються в регістрі символів, переміщається на один розряд праворуч, звільняючи тим самим крайній лівий розряд, куди і надходить новий інформаційний символ [3].

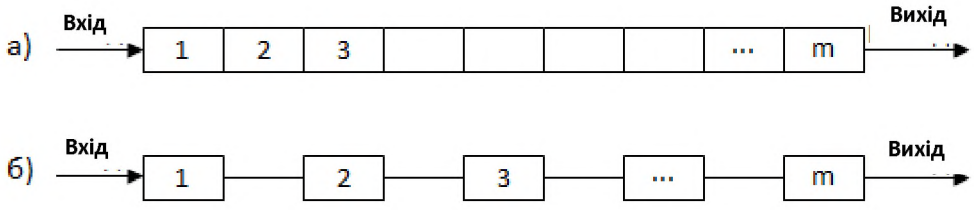

Рисунок 1.13 – Регістр зсуву

Використовуються два різних зображення регістрів зсуву: з зістикованими впритул комірками (рисунок 1.13, а) і з безпосередніми послідовними зв'язками між комірками (рисунок 1.13, б), що дає можливість

на схемах вбудовувати між відповідними комірками додаткові елементи (схеми підсумовування по модулю 2).

Суматор за модулем 2 здійснює складання символів 0 і 1, які надходять на його входи. Правило додавання за модулем 2 наступне: сума двоїчних символів дорівнює 0, якщо число одиниць серед поданих на входи символів парне, і дорівнює 1, якщо це число непарне.

Комутатор здійснює послідовне зчитування символів, що надходять на його входи (контакти) і встановлює на виході черговість посилки кодових символів в канал зв'язку.

За аналогією з блоковими кодами, згорткові коди можна класифікувати на систематичні і несистематичні.

Систематичним згортковим кодом є такий код, для якого в вихідній послідовності кодових символів міститься без зміни послідовність інформаційних символів, що породила її. В іншому випадку згортковий код є несистематичним.

Згорткове кодування створюється проходженням переданої інформаційної послідовності через лінійний зсувний регістр з кінцевим числом станів. Загалом, регістр зсуву складається з M (m-бітових) комірок і лінійного перетворювача, що складається з n функціональних генераторів і виконує алгебраїчні функції, як показано на рисунку 1.14.

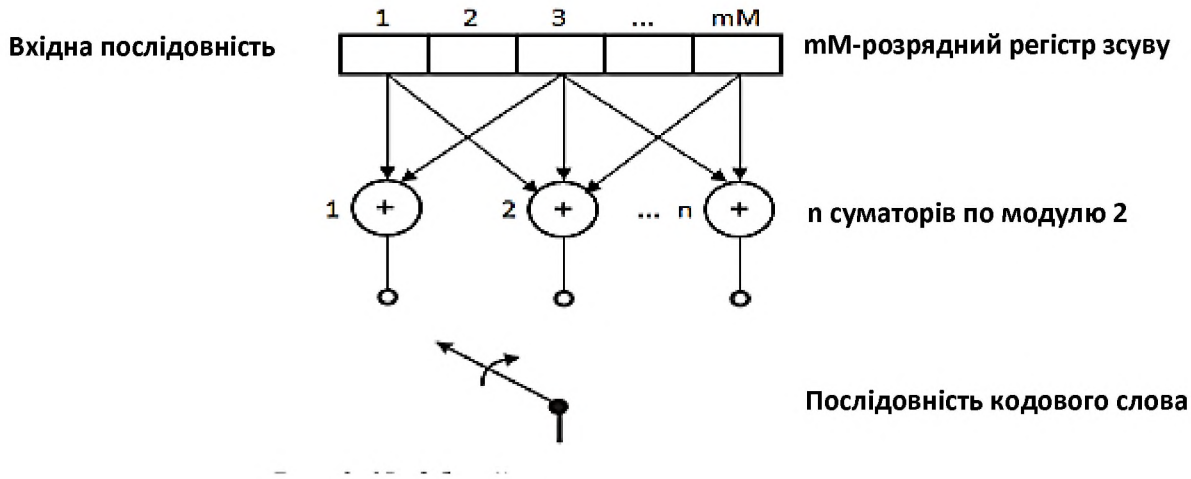

Рисунок 1.14 – Загальний вигляд згорткового кодера

Вхідні дані до кодеру, які вважаються двійковими, просуваються уздовж регістра зсуву по k біт за раз. Число вихідних бітів для кожної k-бітової вхідної послідовності дорівнює n. Отже, кодова швидкість, визначена як $R = k/n$, узгоджується з визначенням швидкості блокового коду.

Методи декодування згорткових кодів

Метод порогового декодування

При пороговому декодуванні згорткових кодів обчислюються синдроми (ознаки місця помилкових символів), потім ці синдроми або послідовності, отримані за допомогою лінійного перетворення синдромів, подаються на входи порогового елемента, де шляхом "голосування" (мажоритарний метод) і порівняння його результатів з порогом виноситься рішення про значення символу, що декодується. Основна перевага цього методу декодування - простота реалізації. Однак він не повністю реалізує потенційні коригувальні здібності згорткових кодів. Крім того, не всі згорткові коди можуть бути декодовані цим методом. Граничне декодування, як правило, застосовується для систематичних кодів [4]. Загальна схема декодера для згорткового коду ($R = 1/2$) представлена на рисунку 1.15.

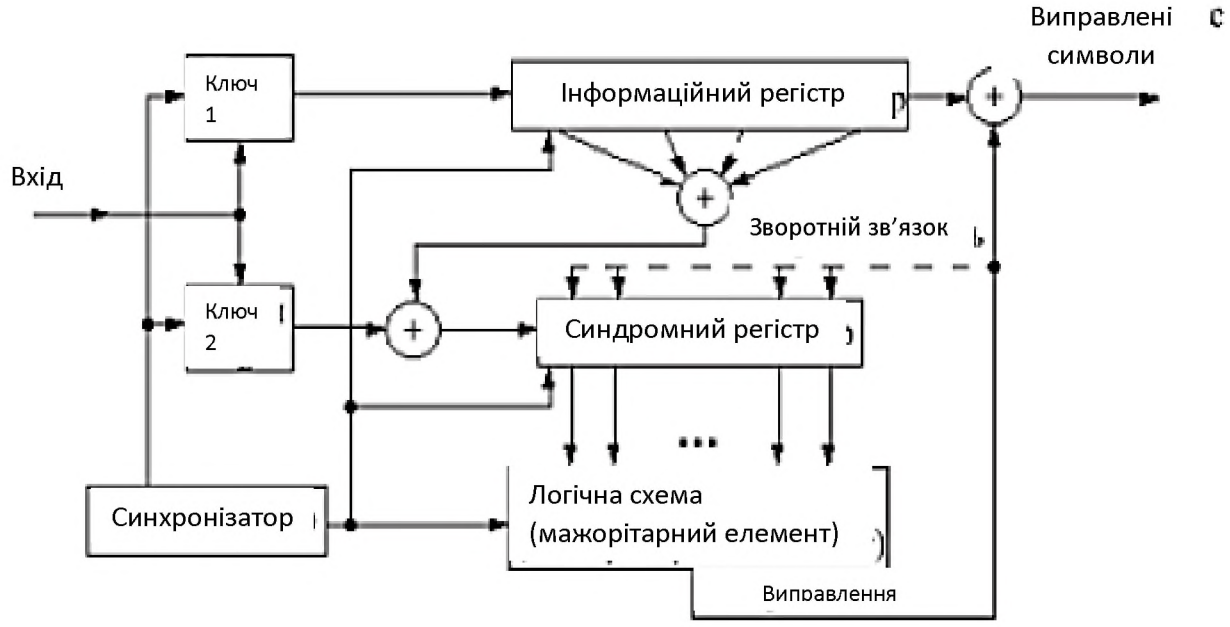

Рисунок 1.15 – Загальна схема декодера для згорткового коду

Декодер містить аналог кодера, в якому за прийнятими інформаційними символами в регістрі зсуву формується копія перевірочної послідовності. З цією метою синхронизатор декодера за допомогою ключів 1 і 2 "розфасовує" вхідну послідовність символів на 2 потоки - інформаційний та перевірки, синхронізатор управляє роботою всього декодера.

У формувачі синдрому (суматорі за модулем 2) утворюється послідовність синдромів S, яка надходить на вхід синдромного регістра. За відсутності в каналі помилок, послідовності на входах формувача синдрому завжди збігаються, і синдромна послідовність складається з одних нулів. Різним наборам помилок відповідають певні конфігурації синдромних послідовностей, в яких на певних позиціях з'являються поодинокі символи. Закон формування перевірочних символів вибирається таким чином, щоб за структурою синдромної послідовності можна було визначити спотворені символи.

Логічна схема визначає по синдрому правильність записаного в інформаційному регістрі блоку інформаційних символів. Якщо є комбінація

помилок, яка може бути виправлена, то логічна схема виправляє помилки в цьому блоці шляхом подачі одиничних символів на вихідну схему підсумовування по модулю 2 в моменти виходу з інформаційного регістра спотворених символів.

Помилки, що виправляються в черговому інформаційному блоці, можуть впливати на символи синдромів, відповідних наступним блокам, оскільки згорткові коди безперервні. Для того щоб декодер зміг повністю реалізувати свої коригувальні можливості, слід виключити вплив цих помилок. Це може бути досягнуто за рахунок зворотного зв'язку, який на схемі (рисунок 1.15) представлений пунктирною лінією.

Зворотній зв'язок перетворює синдромний регістр прямої дії в нелінійний регістр зсуву зі зворотнім зв'язком. Це може привести до явища, званого розмноженням помилок. Невиправні помилки в каналі можуть викликати перехід синдромного регістра в такий стан, що і при відсутності адитивних помилок в каналі декодер буде продовжувати завжди декодувати неправильно. Причина цього полягає в тому, що вихід нелінійного регістра зсуву зі зворотнім зв'язком, коли на його вхід надходить нульова послідовність, а початковий стан - нульовий, може бути періодичним [3].

Послідовний алгоритм декодування

При розгляді алгоритмів послідовного декодування зручно представляти згортковий код у вигляді кодового дерева. Як вже зазначалося раніше, вихідному нульовому стану зсувного регістра кодера відповідає початковий вузол дерева. Якщо вхідний інформаційний символ, що надходить в регістр, дорівнює 1, то йому приписується лінія (ребро дерева), що йде, як прийнято на цьому малюнку, вниз, а якщо інформаційний символ дорівнює 0, - то вгору. Тим самим отримуємо два нових вузла, відповідні наступному такту роботи кодера, для кожного з яких дерево будується далі аналогічним чином, і так далі. Над кожним ребром дерева записуються кодові символи, одержувані при цьому на виході кодера. Сукупність

декількох послідовних ребер, що з'єднують будь-які два вузла, становить гілку дерева.

Кожна послідовність кодованих інформаційних символів породжує певний шлях по кодовому дереву. Очевидно, завдання декодера полягає в знаходженні істинного (правильного) шляху, тобто того шляху, який в дійсності був породжений кодером [4].

Таким чином, при алгоритмах послідовного декодування декодер визначає найбільш правдоподібний шлях по решітці, що дозволяє виключити з аналізу більшу частину решти шляхів, мають меншу правдоподібність.

Для цього, спочатку необхідно пересуватися по черзі вздовж кожного шляху кодової решітки, порівнюючи прийняту послідовність із значенням, відповідним шляху, по якому відбувається рух. Якщо при цьому вдається виявити певний шлях, значення якого збігається з прийнятою послідовністю, то природно вважати, що ця послідовність і передавалася.

Якщо бути точніше, відбувається порівняння прийнятої комбінації з комбінаціями, можливими для даного кроку декодування. Шлях вважається найбільш правдоподібним, якщо метрика між прийнятою послідовністю і послідовністю, що відповідає даному переходу, мінімальна.

Остаточно: на кожному кроці декодування ми маємо два вихідних з вузла шляхи. Спочатку ми обчислюємо відстань Хеммінга між прийнятою комбінацією і комбінацією, що відповідає одному вихідному шляху, потім обчислюємо відстань Хеммінга між прийнятою комбінацією і комбінацією, відповідною другому шляху. Після чого вибираємо той шлях, значення відстані Хеммінга якого менше. Даний процес показаний на малюнку 1.16. В даному випадку перехід вниз по решітці буде вважатися найбільш правдоподібним, так як відстань Хеммінга між значенням, відповідним цьому переходу і прийнятою комбінацією, є найменшою.

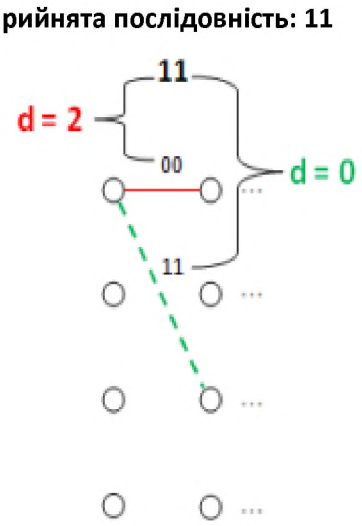

Рисунок 1.16 – Вибір шляху з найменшою відстанню Хеммінга

Декодер виконує операцію вибору шляху з найменшою метрикою до тих пір, поки не зіткнеться з неоднозначністю, при якій прийняти рішення про те, який шлях є найкоротшим, не представляється можливим через те, що відстані Хеммінга між прийнятою послідовністю і значеннями двох можливих шляхів однакові. У цій ситуації декодер змушений перевірити обидва можливі шляхи, і за наступними метриками прийняти рішення стосовно невизначеності.

У нашому ж випадку, якщо відстані Хеммінга для обох шляхів однакові, то вдаємося до жорсткого методу прийняття рішень і користуємося таким правилом:

- Якщо ми знаходимося в вузлі 00 або 11, і прийшла комбінація 00 або 01 - вважаємо що передавався нуль, 10 або 11 - одиниця.

- Якщо ми знаходимося в вузлі 01 або 10, і прийшла комбінація 00 або 01 - вважаємо, що передавалася одиниця, 10 або 11 - нуль.

Аналогічним чином декодер продовжує заглиблюватися в решітку і приймати рішення, що стосуються інформаційних бітів, усуваючи всі шляхи, окрім одного.

На рисунку 1.17 зображена ситуація, коли на вхід декодера надходить послідовність, в якій біти, що виділені кольором, свідомо спотворені так, як це буває в реальних каналах зв'язку.

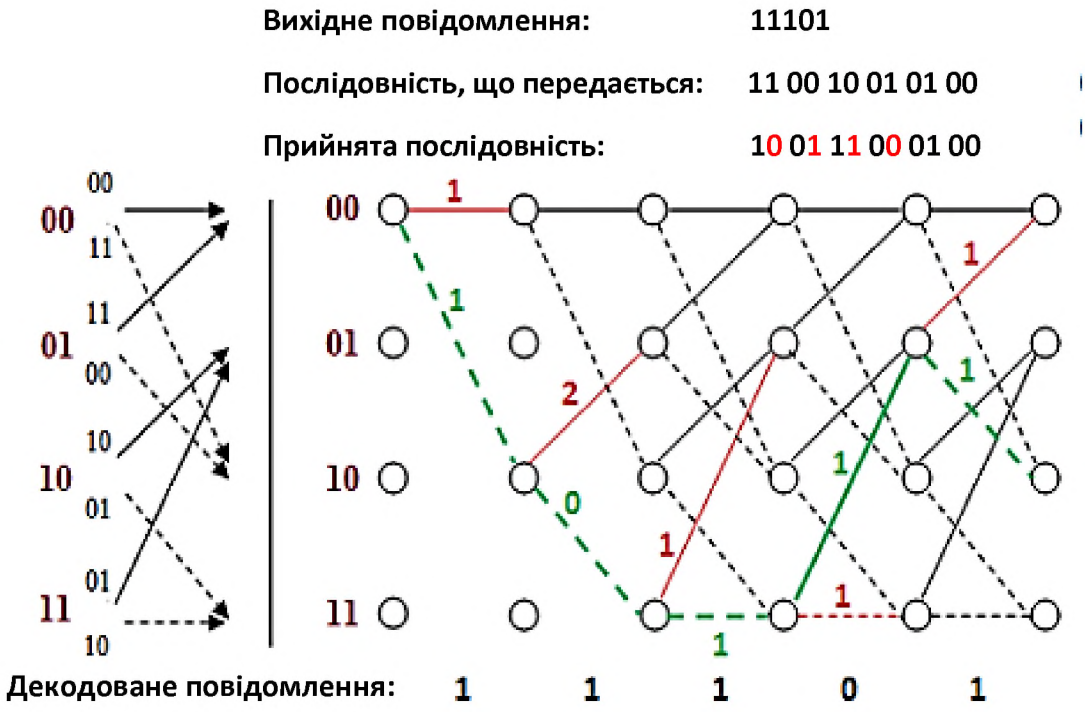

Рисунок 1.17 – Решітчаста діаграма послідовного декодера

Порівняння алгоритму Вітербі з послідовним алгоритмом

Принципова відмінність алгоритму Вітербі від послідовного алгоритму декодування полягає в тому, що алгоритм Вітербі приймає рішення по сумарним метрикам шляхів, причому в вузлах, де ці шляхи сходяться. Цей крок в процесі декодування показаний на рисунку 1.18 (метрики позначені даним чином для наочності).

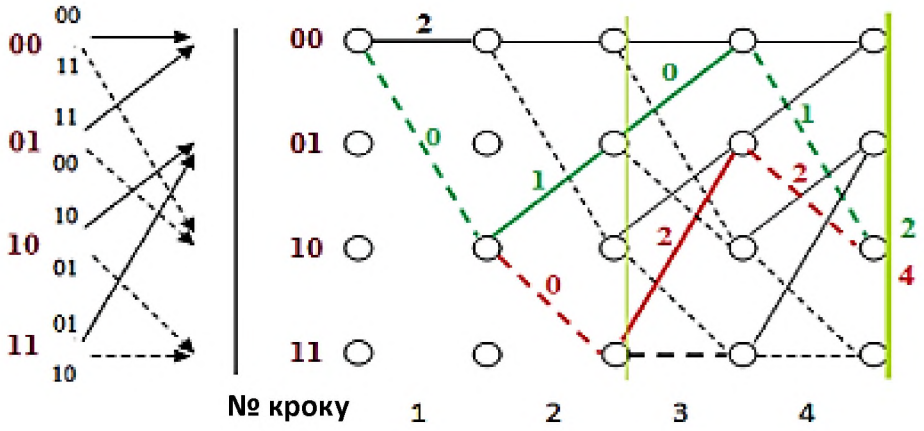

Рисунок 1.18 – Решітчаста діаграма декодера Вітербі

Таким чином, якби в ситуації, представленій на рисунку 1.17, ми приймали рішення на основі послідовного алгоритму, то на першому і другому кроці декодування ми прийняли б рішення йти вниз по решітці, вибираючи найкоротші метрики Хеммінга. Однак в такому випадку ми б не врахували метрики наступних гілок нашого, на перший погляд найкоротшого шляху, які на кроці три і чотири сильно зростають, а значить, вибрали б не найкоротший шлях при декодуванні. Тому алгоритм Вітербі є оптимальним. Однак він складніше послідовного як в розумінні, так і в реалізації. В якості метрики виступає відстань Хеммінга.

1.3 Постановка задачі

Метою дипломної роботи є дослідження характеристик кодеків шляхом імітаційного моделювання.

Для реалізації поставленої мети необхідно вирішити наступні завдання:

1 Виконати аналітичний огляд методів побудови найбільш широко застосовуваних кодеків.

2 Розробити імітаційні моделі найбільш широко застосовуваних кодеків.

3 Виконати модельний експеримент з метою дослідження характеристик кодеків.

4 Виконати аналіз отриманих результатів.

1.4 Висновки

1 Виконано аналітичний огляд методів побудови найбільш широко застосовуваних кодеків.

2 Виконано класифікація коригувальних кодів.

3 Виконано аналітичний огляд основних коригувальних кодів

4 Виконано постановка задачі дослідження.

2 СПЕЦІАЛЬНА ЧАСТИНА

2.1 Моделювання кодека Хеммінга і дослідження його характеристик

У середовищі Simulink MatLab необхідно зібрати схему для роботи кодека Хеммінга (7,4). Схема представлена на рисунку 2.1.

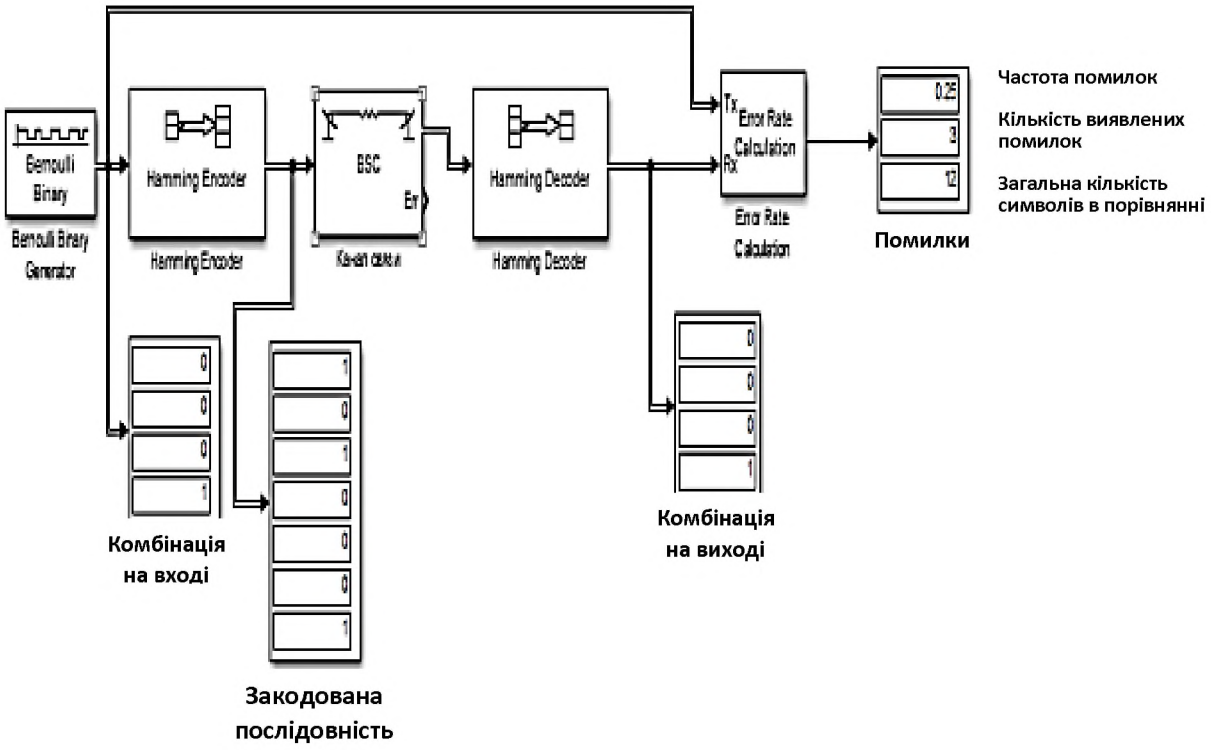

Рисунок 2.1 – Модель системи передачі з застосуванням коду Хеммінга

До складу системи входять:

1 Bernoulli Binary Generator

2 Hamming Encoder

3 Binary Symmetric Channel (канал передачі)

4 Hamming Decoder

5 Error Rate Calculation (аналізатор помилок)

6 Display

Після установки характеристики блоків системи для коду (7,4) і проведення модельного експерименту отримуємо результати, наведені на рис. 2.2 ... 2.5.

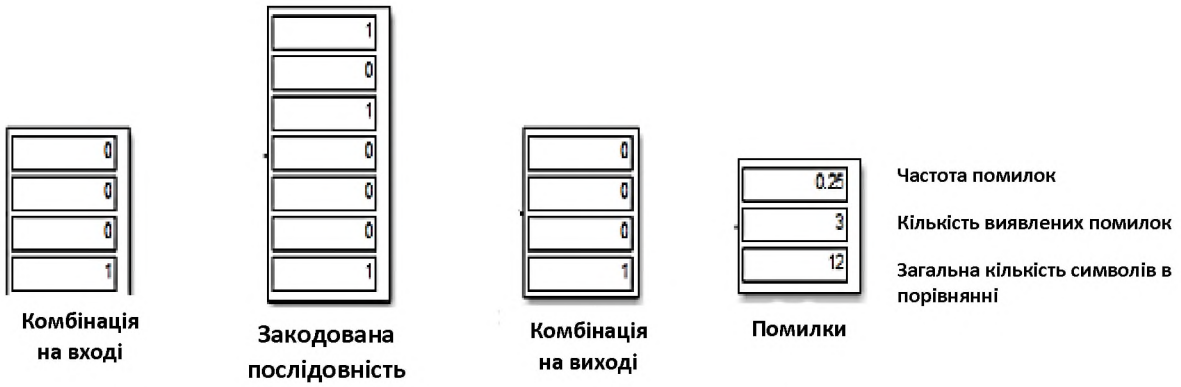

Рисунок 2.2 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,2)

Аналізуючи рисунок 2.2, можна зробити висновок, що комбінація на вході збігається з комбінацією на виході, таким чином, передача здійснилася вдало. Що стосується помилок, то їх частота дорівнює 0,25, число виявлених помилок дорівнює 3, загальна кількість символів в порівнянні дорівнює 12. Кодування та декодування тут здійснюється методом, описаним в розділі 1.

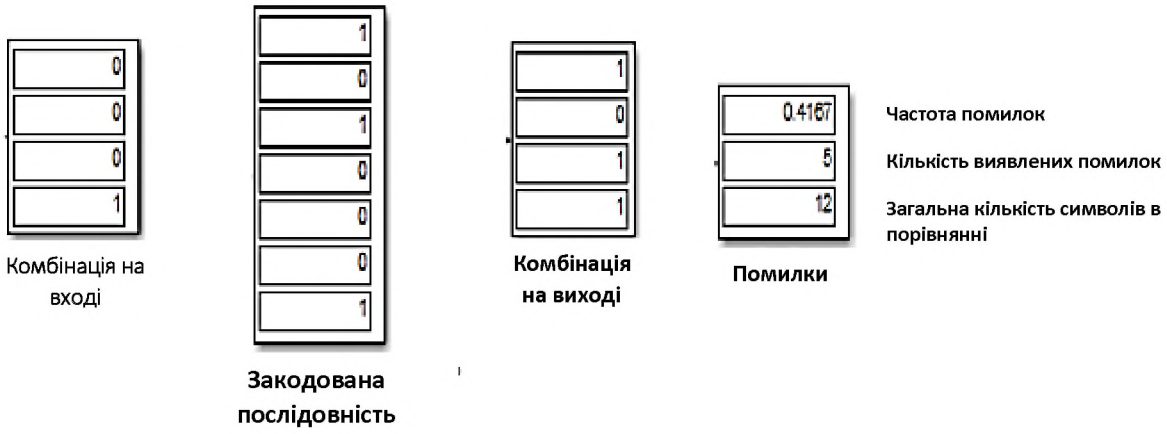

Рисунок 2.3 Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,4)

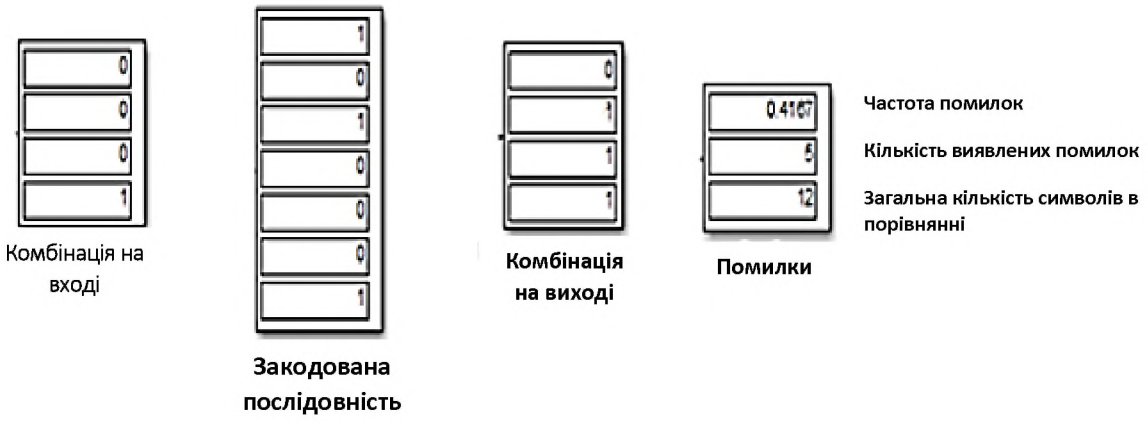

Рисунок 2.4 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,6)

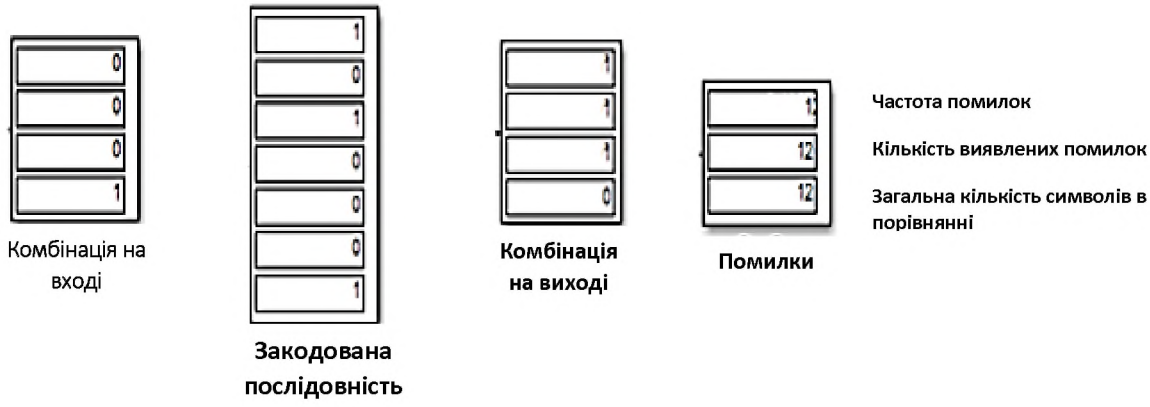

Рисунок 2.5 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,8)

На підставі отриманих даних побудуємо залежність числа помилок від ймовірності помилки в каналі зв'язку для коду Хеммінга (7,4) (рис. 2.6).

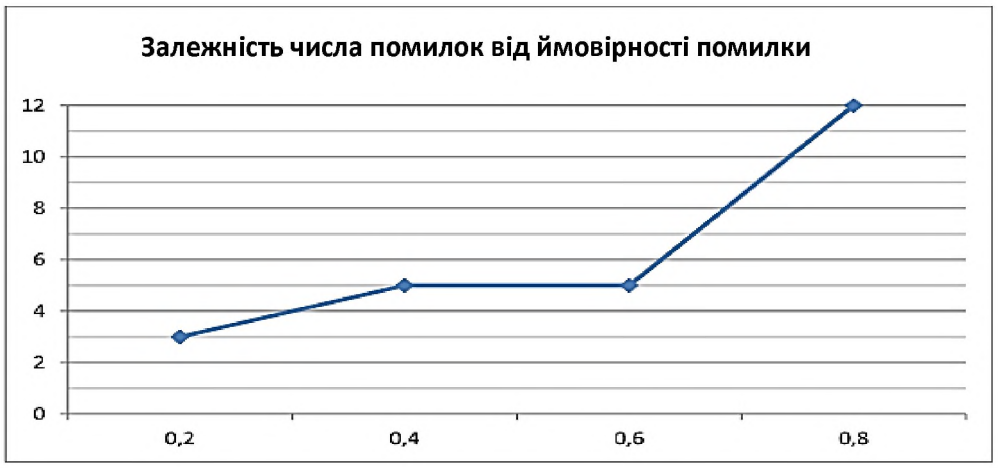

Рисунок 2.6 – Графік залежності числа помилок (вісь OY) від ймовірності помилки (вісь OX) для коду Хеммінга (7,4)

Аналогічна залежність може бути побудована для коду Хеммінга (15, 11) (рис. 2.7).

Рисунок 2.7 – Графік залежності числа помилок (вісь OY) від ймовірності помилки (вісь OX) для коду Хеммінга (15,11)

В результаті моделювання побудовано схему системи передачі з кодуванням Хеммінга в середовищі Simulink. Побудовано графіки залежностей числа помилок на виході декодера від ймовірності помилки в каналі зв'язку для кодів (7,4) і (15,11).

З графіків (рисунок 2.6 і рисунок 2.7) видно, що число помилок збільшується зі зростанням ймовірності помилок в каналі зв'язку.

2.2 Моделювання кодека БЧХ та дослідження його характеристик

У середовищі Simulink MatLab необхідно зібрати схему для роботи кодека БЧХ (15, 7). Схема представлена на рисунку 2.8.

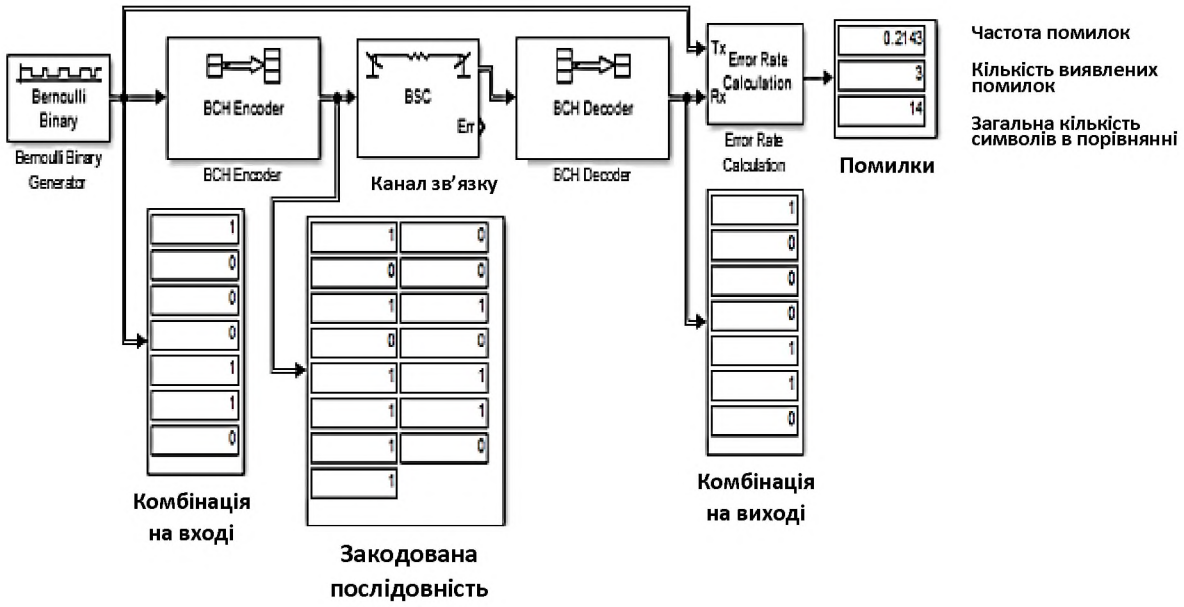

Рисунок 2.8 – Модель системи передачі з застосуванням коду БЧХ

До складу системи з кодуванням входять:

1. Bernoulli Binary Generator
2. BCH Encoder
3. Binary Symmetric Channel (канал передачі)
4. BCH Decoder
5. Error Rate Calculation (аналізатор помилок)
6. Display

Після встановлення характеристик блоків схеми рис. 2.8 для коду (15, 7) і проведення модельного експерименту отримуємо результати, які проілюстровані рис. 2.9 ... 2.12.

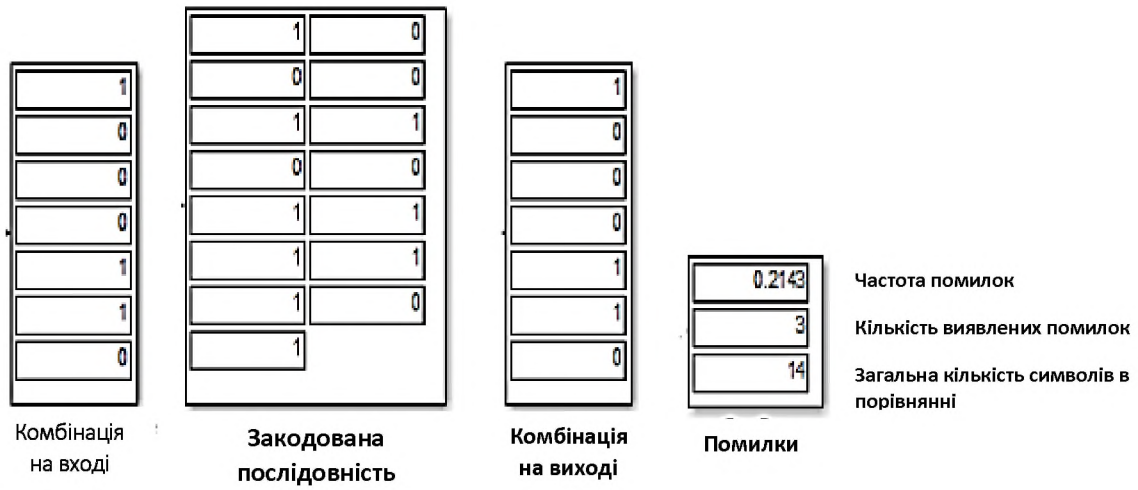

Рисунок 2.9 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,2)

З рис. 2.9 можна зробити висновок, що комбінація на вході збігається з комбінацією на виході, таким чином, передача здійснилася вдало. Що стосується помилок, то їх частота дорівнює 0,2143, число виявлених помилок дорівнює 3, загальна кількість символів в порівнянні рівна 14. Кодування та декодування тут здійснюється методом, який описаний вище.

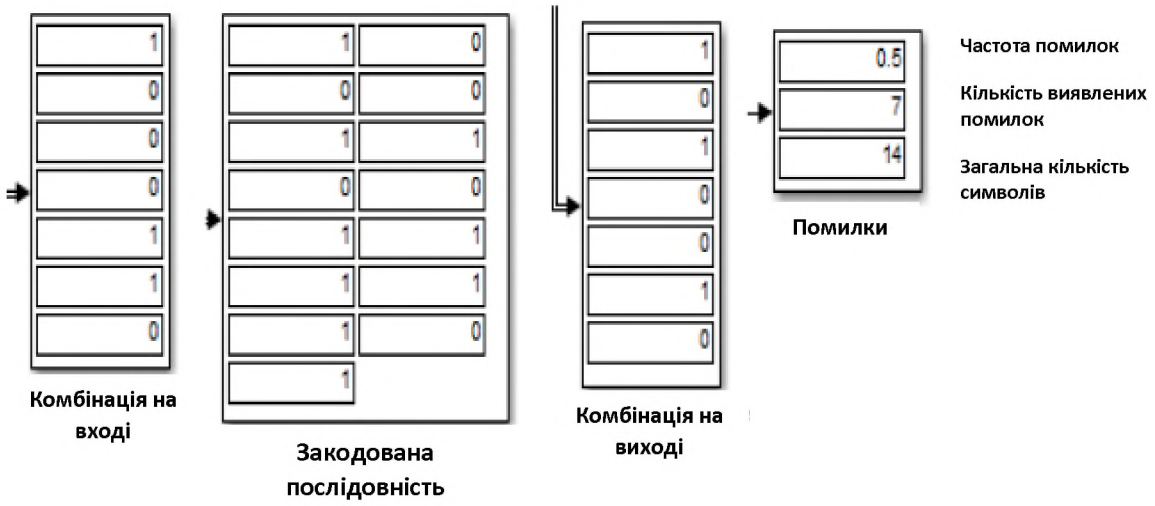

Рисунок 2.10 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,4)

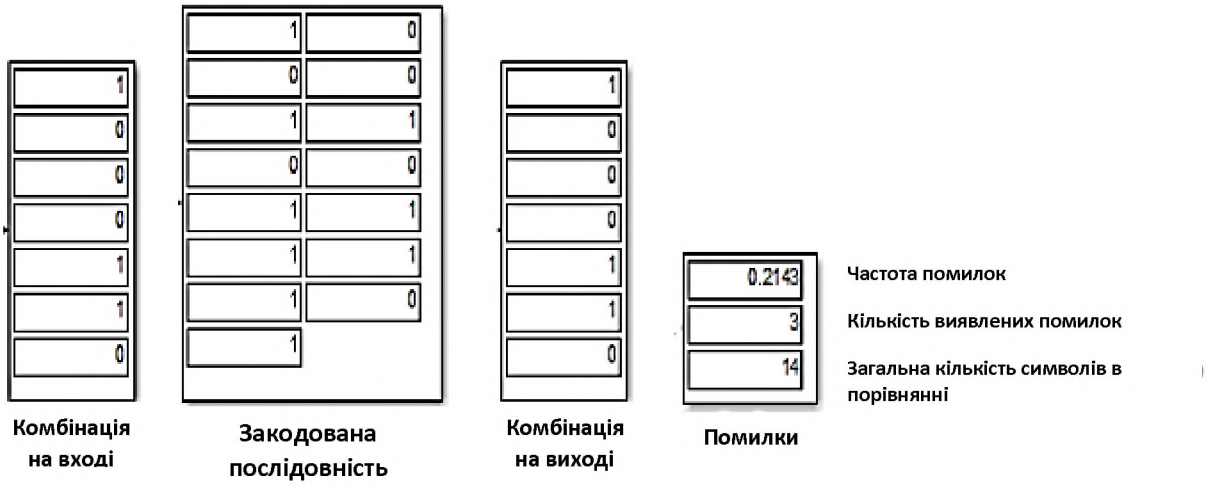

Рисунок 2.11 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,6)

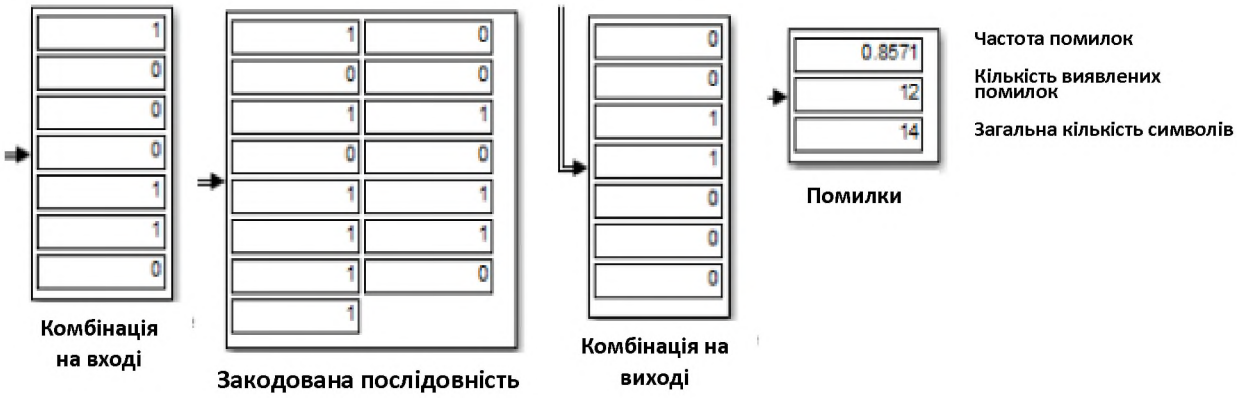

Рисунок 2.12 – Комбінація на вході, закодована послідовність, комбінація на виході, помилки (ймовірність помилок дорівнює 0,8)

Встановлюємо характеристики блоків схеми для коду (15,11) і отримуємо аналогічні результати. Графіки, що ілюструють отримані результати, наведені на рис. 2.13 - 2.14.

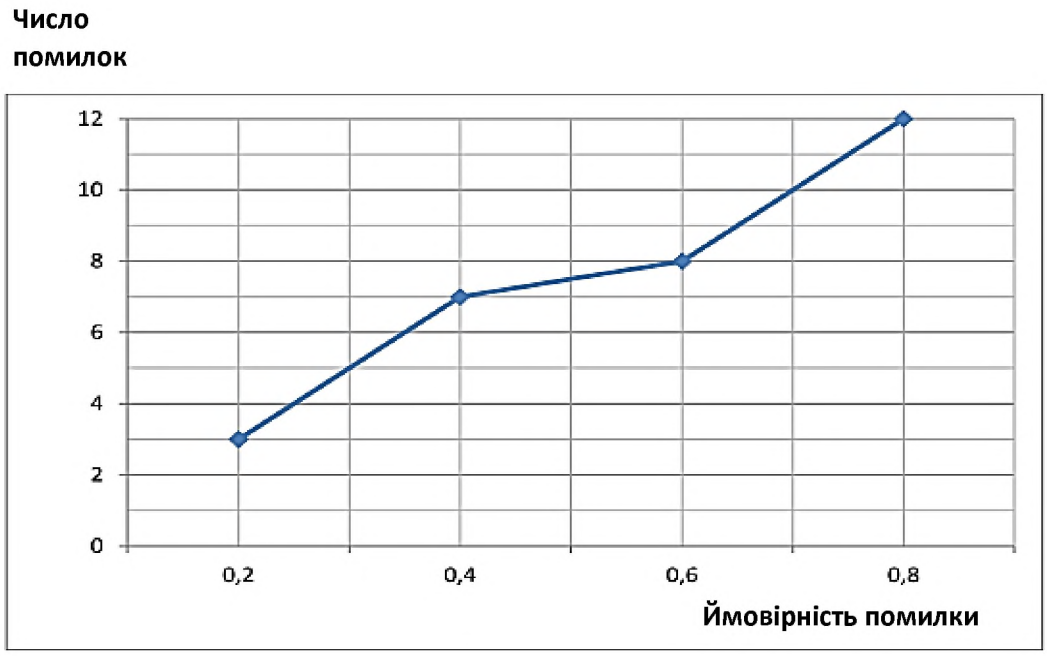

Рисунок 2.13 – Графік залежності числа помилок від ймовірності помилки для

коду БЧХ (15,7)

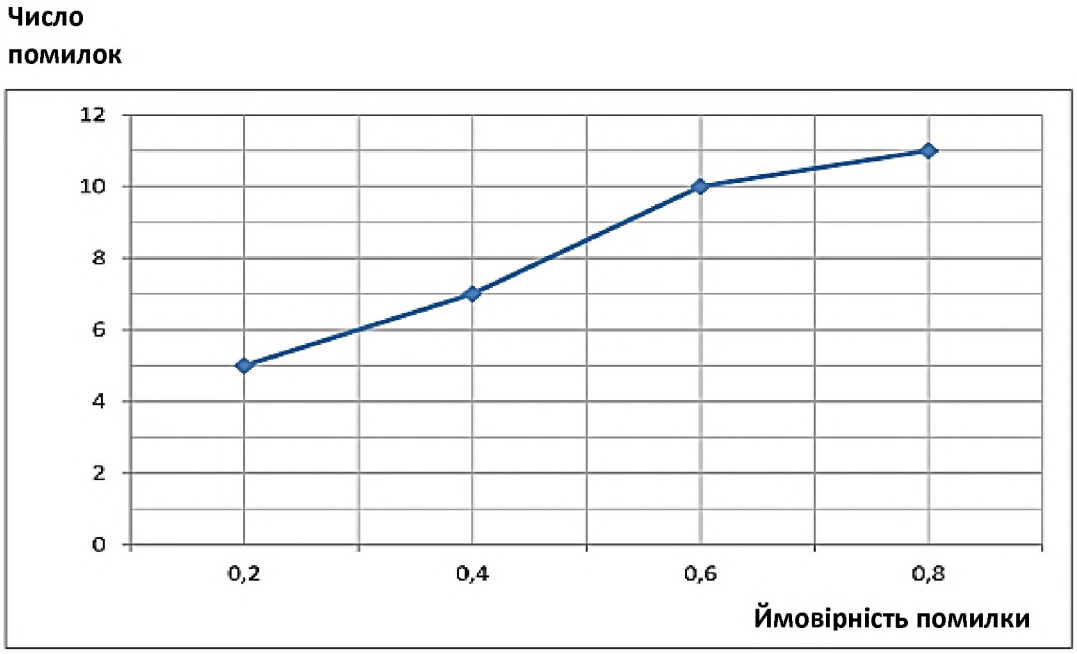

Рисунок 2.14 – Графік залежності числа помилок від ймовірності помилки для

коду БЧХ (15,11)

В результаті модельного експерименту з використанням розробленої моделі системи передачі з кодуванням БЧХ в середовищі Simulink, побудовані графіки залежностей числа помилок на виході декодера від ймовірності помилки в каналі зв'язку для кодів (15,7) і (15,11). З графіків (рисунок 2.13 і рисунок 2.14) видно, що число помилок збільшується зі зростанням ймовірності помилок. При цьому код (15,7) виявився краще за здатністю виявлення помилок.

2.3 Моделювання кодека Ріда-Соломона і дослідження його характеристик

На даному етапі необхідно реалізувати і проаналізувати описані вище алгоритми кодування на практиці в програмному середовищі MatLab. Структурна схема системи аналогічна розглянутим раніше, за винятком того, що були обрані відповідні блоки кодера і декодера.

Отримані результати наведені на рис. 2.15 ... 2.17.

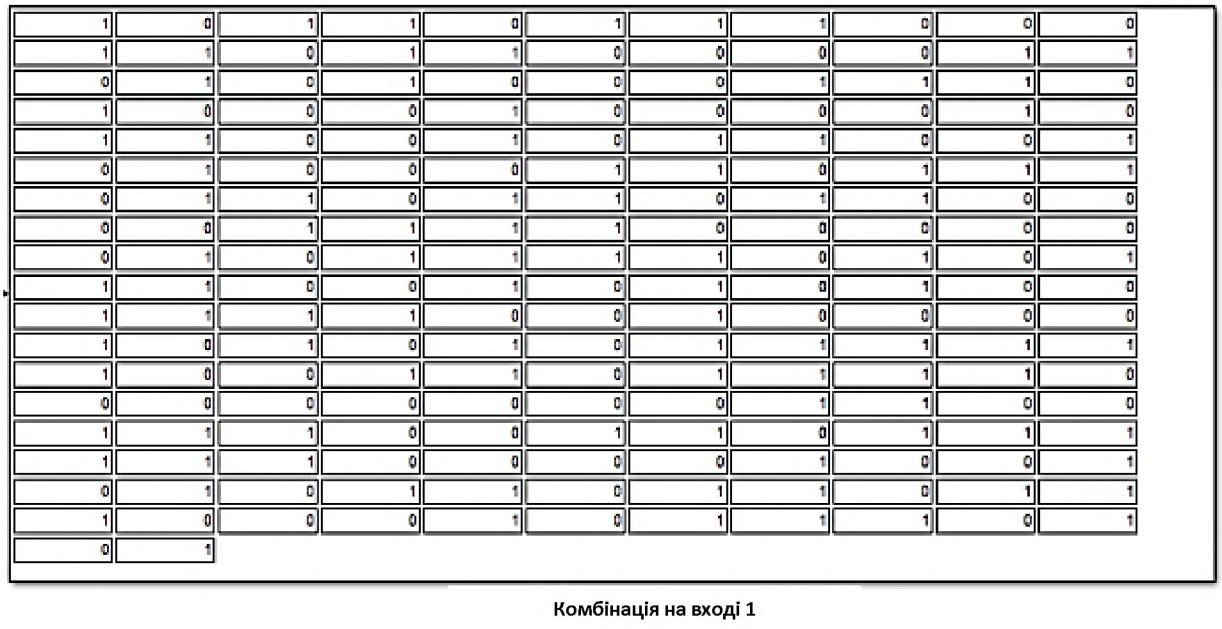

Рисунок 2.15 – Вхідна послідовність

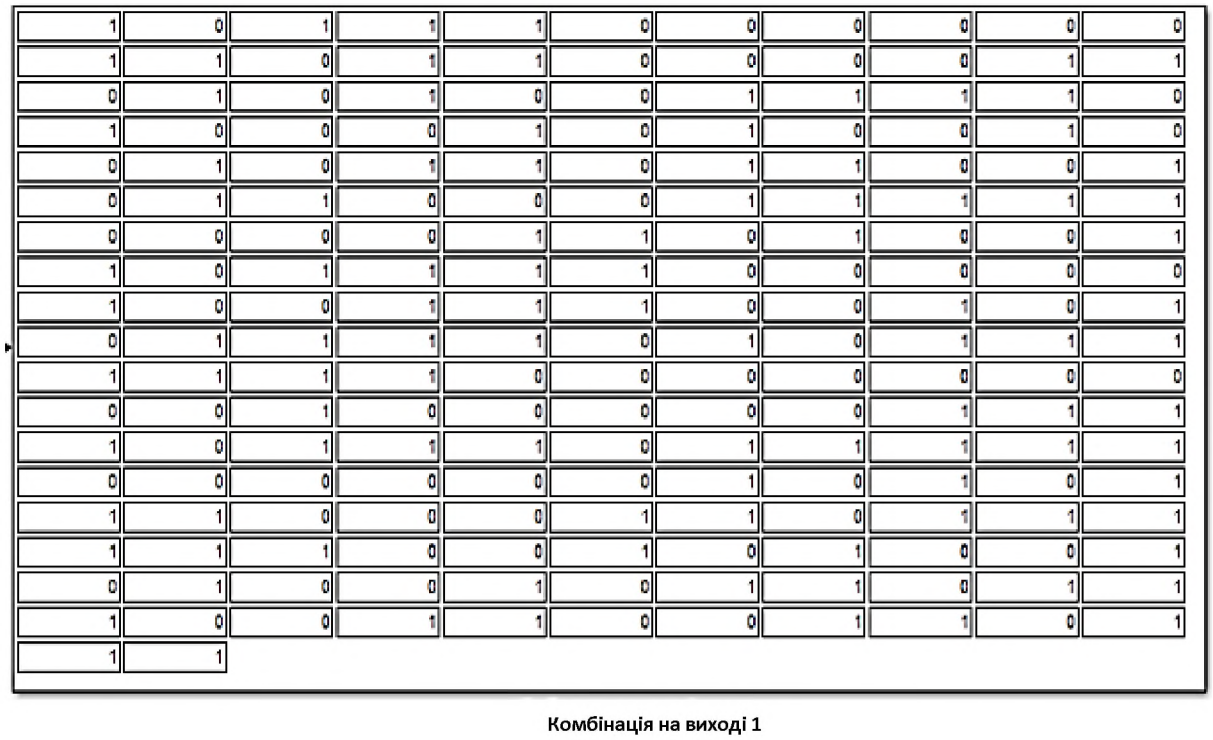

Рисунок 2.16 – Вихідна послідовність

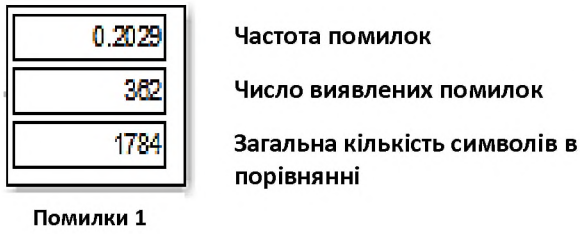

Рисунок 2.17 – Лічильник помилки

Аналізуючи рисунки 2.15 ... 2.17, можна зробити висновок, що комбінація на вході збігається з комбінацією на виході, таким чином, передача здійснилася вдало. Що стосується помилок, то їх частота дорівнює 0,2029, число виявлених помилок дорівнює 362, загальна кількість символів в порівнянні дорівнює 1784. При збільшенні ймовірності помилки до 0,3 в декодованій послідовності були знайдені помилки.

Графік залежності числа виявлених помилок від ймовірності помилки для коду (255,223) наведено на рис. 2.18.

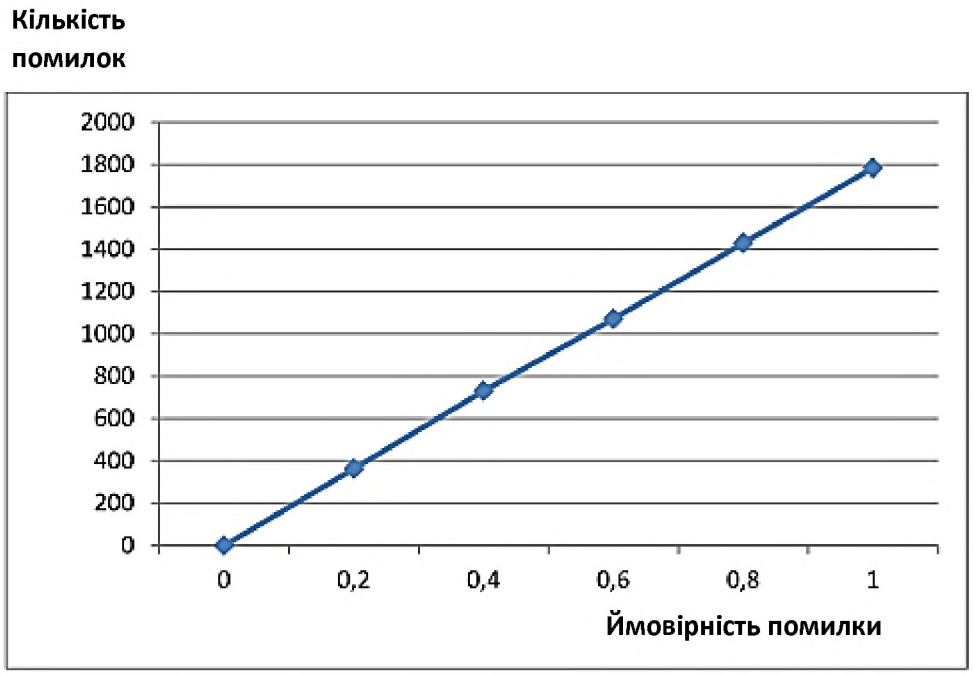

Рисунок 2.18 – Графік залежності числа виявлених помилок від ймовірності помилки для коду (255,223)

Були розглянуті особливості коду Ріда-Соломона. Побудувані графіки залежності числа виявлених помилок від ймовірності помилки для коду (255,223). Встановлено:

1 Код Ріда-Соломона - код з 8-бітними символами і перевіркою на парність. У коді (255,223) 223 байта інформаційних символів, 32 байта перевірки на парність.

2 Обсяг обчислювальної потужності, необхідної для кодування і декодування кодів Ріда-Соломона, залежить від числа символів парності. Велике значення t означає, що більша кількість помилок може бути виправлена, але це зажадає більшої обчислювальної потужності в порівнянні з варіантом при меншому t.

3 Код RS (255,223) може виправити до 16 помилок в символах. У гіршому випадку, можуть мати місце 16 бітових помилок в різних символах (байтах). У кращому випадку, коригуються 16 повністю невірних байт, при цьому виправляється 16 * 8 = 128 бітових помилок.

Таблиця 2.1 Результати

	Ймовірність помилки 0,2	**Ймовірність помилки 0,3**
Код Хемминга (7,4)	**Частота помилок: 0,25** **Число виявлених помилок: 3** **Загальна кількість символів: 12**	**Частота помилок: 0,4167** **Число виявлених помилок: 5** **Загальна кількість символів: 12**
Код БЧХ (15,7)	**Частота помилок: 0,2143** **Число виявлених помилок: 3** **Загальна кількість символів: 14**	**Частота помилок: 0,4286** **Число виявлених помилок: 6** **Загальна кількість символів: 14**
Код Р-С (255,223)	**Частота помилок: 0,2029** **Число виявлених помилок 362** **Загальна кількість символів: 1784**	**Частота помилок: 0,31** **Число виявлених помилок: 553** **Загальна кількість символів: 1784**

2.4 Моделювання CRC-кодека

Модель передачі даних з виявленням помилок за допомогою CRC- коду була реалізована в середовищі Simulink MatLab. Модель демонструє роботу CRC-кодера і декодера, дозволяє досліджувати виявляючу здатність коду для різних генераторних поліномів.

На рис. 2.19 наведена розроблена модель:

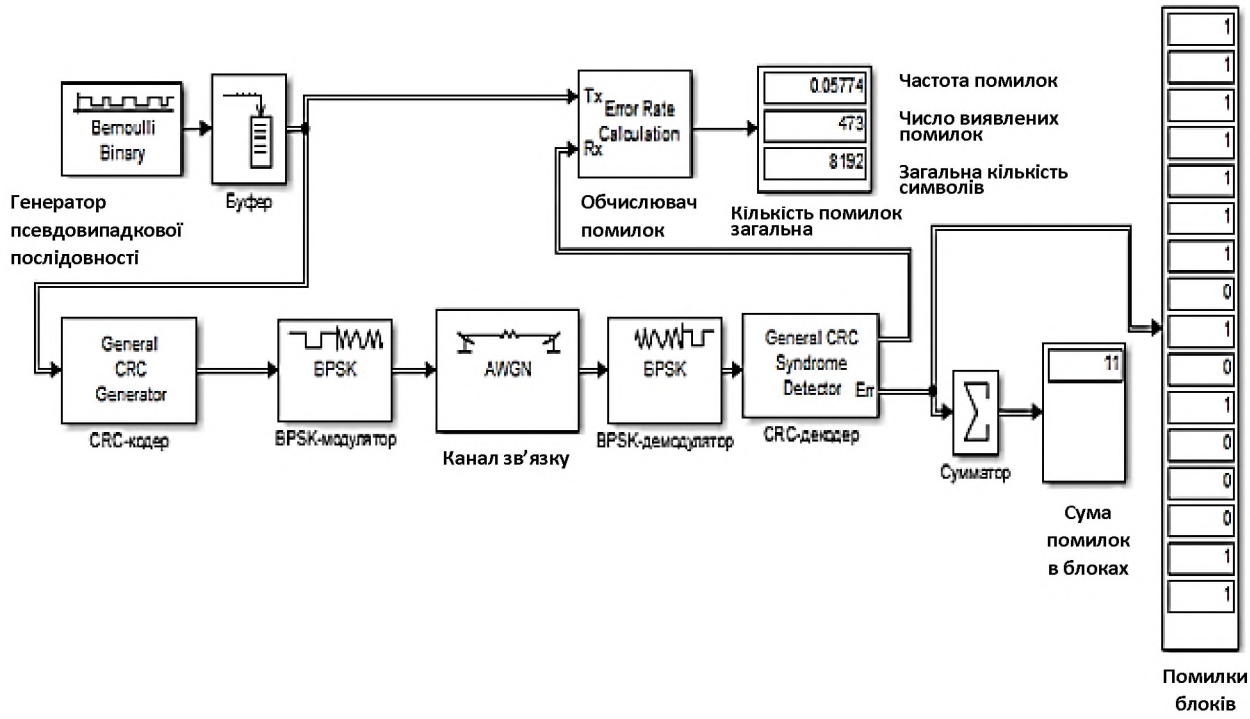

Рисунок 2.19 Модель дослідження CRC-кодів

Модель циклічного надлишкового коду, представлена на рисунку 2.19, дозволяє досліджувати виявляючу здатність CRC кодів з різними поліномами.

Задаємо однаковий генераторний поліном в блоки CRC-кодер і CRC-декодер.

Загальна кількість переданих символів становить 8192. Кількість контрольних сум змінюється від 2 до 8192, зі збільшенням кожного попереднього значення в 2 рази (2, 4, 8, 16 ... 8192).

Значення SNR в блоці «Про з'єднання» встановлено в 1 дБ. Таким чином, бітова ймовірність помилки (BER) складе 0,05786.

На рис. 2.20 представлений графік залежності числа виявлених помилок від кількості контрольних сум для різних поліномів CRC-коду.

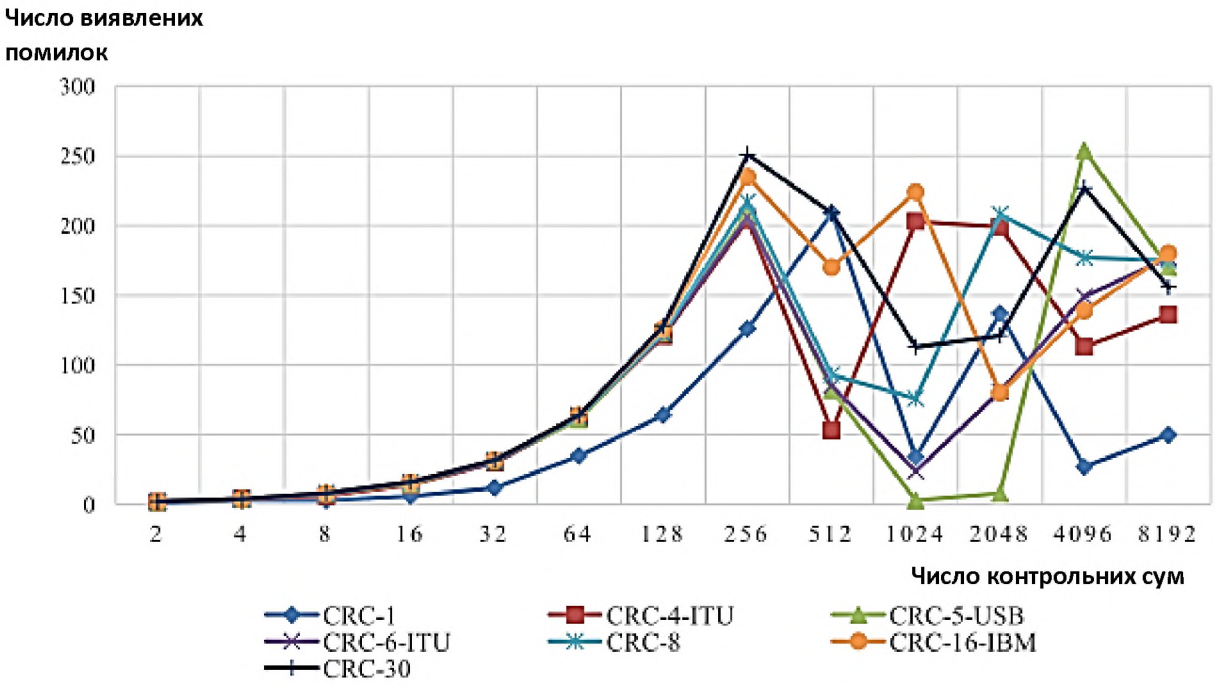

Рисунок 2.20 – Графік залежності числа виявлених помилок від кількості контрольних сум для різних поліномів CRC-коду

У таблиці 2.2 наведені найбільш поширені поліноми CRC кодів.

У цьому параграфі проведено дослідження моделі циклічного надлишкового коду (CRC).

Модель дозволяє досліджувати CRC-коди з можливістю завдання будь-якого генераторного полінома і зміні кількості контрольних сум у фреймі.

Отримані наступні результати і зроблені наступні висновки:

1) чим вище ступінь полінома, тим краще його виявляюча здатність;

2) для кожного полінома є таке число контрольних сум в блоці, при якому його виявляюча здатність максимальна, причому у всіх поліномів ці точки різні.

.

Однак, при виборі полінома CRC-коду також необхідно враховувати й інші фактори:

1) збільшення ступеня полінома призводить до ускладнення реалізації кодера і декодера;

2) чим вище частота обчислення контрольних сум, тобто чим більше контрольних сум додається до блоку даних, тим менше пропускна здатність каналу;

3) CRC-коди використовують для виявлення помилок, що означає наявність каналу перезапитання.

При виборі між кодом CRC-каналом перезапитання і перешкодостійким кодуванням, необхідно враховувати характеристики каналу зв'язку. При великому числі помилок передача даних буде неможлива.

4) Вибір полінома залежить від розміру переданого блоку даних, чим більше блок - тим вище ступінь поліному необхідно підбирати. Таким чином, існує обмеження на розмір блоку даних, інакше в будь-якому блоці на приймальному кінці буде виявлятися помилка.

Таблиця 2.2. Поширені поліноми CRC кодів

Назва	Поліном
CRC-1	$x+1$ (використовується в апаратному контролі помилок, також відомий як біт парності)
CRC-4-ITU	x^4+x+1
CRC-5-ITU	$x^5+x^4+x^2+1$
CRC-5-USB	x^5+x^2+1
CRC-6-ITU	x^6+x+1
CRC-7	x^7+x^3+1 (Системи телекомунікації , ITU-T G.707, ITU-T G.832, MMC, SD)
CRC-8	$x^8+x^7+x^6+x^4+x^2+1$
CRC-16-IBM	$x^{16}+x^{15}+x^2+1$ (Bisync, Modbus, USB, ANSI X3.28)
CRC-16-CCITT	$x^{16}+x^{12}+x^5+1$ (X.25, HDLC, XMODEM, Bluetooth, SD)
CRC-30	$x^{30}+x^{29}+x^{21}+x^{20}+x^{15}+x^{13}+x^{12}+x^{11}+x^8+x^7+x^6+x^2+x+1$ (CDMA)

2.5 Моделювання кодека на базі згорткового кодування

Програмна модель, розроблена з використанням підсистеми візуального моделювання Simulink, дозволяє провести дослідження якісних характеристик системи зв'язку з використанням згорткового кодування при «м'якому» і «жорсткому» рішенні.

Модель макета в MatLab / Simulink для згорткового декодування Вітербі при «М'якому рішенні» представлена на рисунку 2.21.

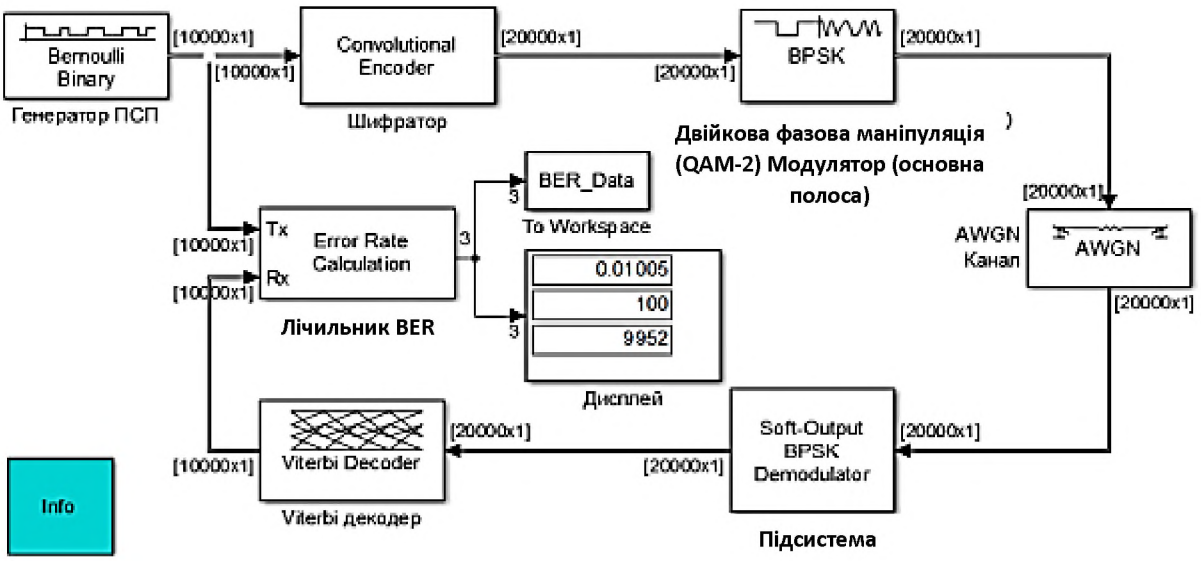

Рисунок 2.21 - Модель згорткового декодера Вітербі при «М'якому рішенні»

Модель макету в MatLab/Simulink для згорткового декодування Вітербі при «Жорсткому рішенні» представлена на рис. 2.22.

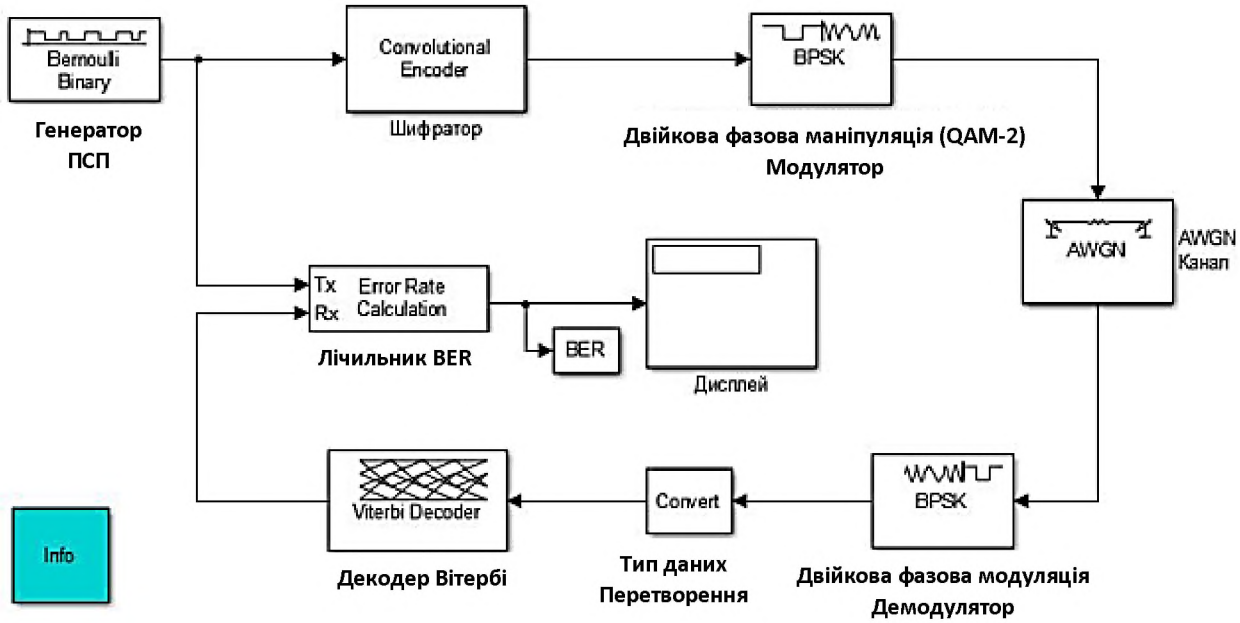

Рисунок 2.22 – Модель згорткового декодера Вітербі при «Жорсткому рішенні»

Отримані результати

На рис. 2.23 представлений графік залежності коефіцієнта бітових помилок від відношення сигналу до шуму для жорсткого і м'якого прийнять рішень згорткового декодування послідовного методу Вітербі.

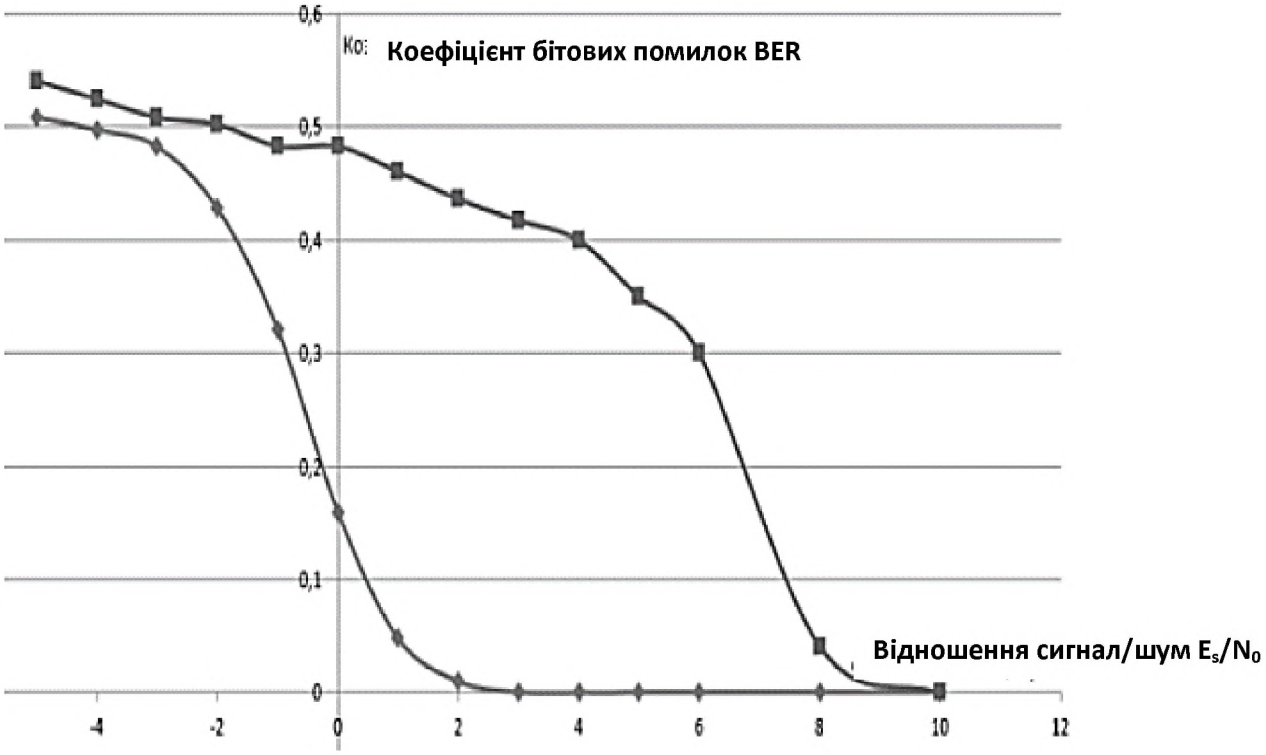

Рисунок 2.23 - Графік залежності коефіцієнта бітових помилок від відношення сигналу до шуму для жорсткого (верхня лінія) і м'якого (нижня лінія) прийняття рішень згорткового декодування послідовного методу Вітербі

З рисунка 2.23 графіка $BER = f\left(E_s / N_0\right)$ видно, що при «м'якому рішенні» декодування проходить краще, ніж при «жорсткому рішенні» тобто

при «м'якому вирішенні» кількість бітових помилок менше, ніж при «жорсткому рішенні».

2.6 Висновки

1 Виконано моделювання кодека Хеммінга і дослідження його характеристик.

2 Виконано моделювання кодека БЧХ і дослідження його характеристик.

3 Виконано моделювання кодека Ріда-Соломона і дослідження його характеристик.

4 Виконано моделювання CRC-кодека і дослідження його характеристик.

5 Виконано моделювання кодека на базі згорткового кодування і дослідження його характеристик.

6 Виконано дослідження та аналіз характеристик імітаційних моделей кодеків.

3 ЭКОНОМИЧЕСКАЯ ЧАСТЬ

В цьому проекті розроблено імітаційні моделі кодеків. У економічному розділі розраховуються одноразові капітальні витрати на розробку моделів кодеків.

3.1.1 Визначення трудомісткості розробки моделей

Трудомісткість створення моделей визначається тривалістю кожної робочої операції, починаючи зі складання технічного завдання і закінчуючи оформленням документації (за умови роботи одного проектувальника):

$$t = t_{mз} + t_{в} + t_{а} + t_{пр} + t_{опр} + t_{д} \text{[год]}. \quad (3.1)$$

де $t_{mз}$ – тривалість складання технічного завдання на впровадження методу;

$t_{в}$ – тривалість вивчення технічного завдання (ТЗ) та літературних джерел за темою;

$t_{а}$ – тривалість розробки моделі;

$t_{пр}$ – тривалість модулювання віртуального аналога каналу зв'язку;

$t_{опр}$ – тривалість опрацювання здобутих характеристик;

$t_{д}$ – тривалість підготовки технічної документації.

Вихідні дані для визначення трудомісткості створення моделей приведені в таблиці 3.1.

Таблиця 3.1 – Тривалість розробки моделей

$tmз$, год	$tв$, год	ta, год	$tпр$, год	$tопр$, год	$tд$, год
50	55	70	20	20	32

Розрахуємо трудомісткість розробки моделей за формулою (3.1):

$$t = 50+55+70+20+20+32=247[год]. \quad (3.1)$$

3.1.2 Розрахунок витрат на розробку моделей

Витрати на розробку моделей *Кпз* складаються з витрат на заробітну платню розробника *Ззп* і вартості витрат машинного часу, що необхідний для опрацювання моделі мережі на ПК *Змч*:

$$К_{пз} = З_{зп} + З_{мч}[грн] \quad (3.2)$$

Заробітна плата виконавця враховує основну і додаткову заробітну плату, а також відрахування на соціальні потреби (пенсійне страхування,

страхування на випадок безробіття, соціальне страхування тощо) і визначається за формулою:

$$З_{зп} = t \cdot З_{пр} [грн]. \quad (3.3)$$

де *t*– трудомісткість створення моделі;

Зпр дорівнює 80 грн/год.

Розрахуємо заробітну платню проектувальника за формулою (3.3):

$$З_{зп} = 247 \cdot 80 = 19760 [грн].$$

Вартість машинного часу на ПК визначається за формулою:

$$Змч = \left(та + tпр + tопр + tд \right) \cdot Смч [грн]. \quad (3.4)$$

де $С_{мч}$– вартість 1 години машинного часу ПК, грн/година.

Вартість 1 години машинного часу ПК визначається за формулою:

$$Смч = Р_е \cdot t \cdot С_е + \frac{Ф_{перв} \cdot Н_а}{F_р} + \frac{К_{лпз} \cdot Н_{апз}}{F_р} [грн/год], \quad (3.5)$$

де *Ре* – встановлена потужність ПК;

t– трудомісткість створення моделі;

Се – енерговитрати;

Фперв – первісна вартість ПК на початок року;

На –річна норма амортизації на ПК;

Клпз – вартість ліцензійного програмного забезпечення;

Напз – річна норма амортизації на ліцензійне програмне забезпечення;

Fр –річний фонд робочого часу (за 40-годинного робочого тижня).

Енерговитрати розраховуються за формулою:

$$C_e = P_e \cdot C_{кВт} \text{ [грн/год]}, \tag{3.6}$$

де $C_{кВт}$ - тариф на електричну енергію.

Розрахунок витрат на розробку моделей зводимо в таблицю 3.2

Таблиця 3.2 – Розрахунок витрат на розробку моделей

Ре, кВт	$C_{кВт}$ кВт·год	*Фперв*, грн	*На*, частка одиниці	*Клпз*, грн	*Напз*, частка одиниці	*Fр*, год
1,0	1,60	18000	0,4	8000	0,4	1920

Тоді за формулою (3.6) отримаємо розмір енерговитрат:

$$Се = 1{,}0 \cdot 1{,}60 = 1{,}60 \text{ [грн/год]}.$$

Річна норма амортизації, якщо використовується метод прискорення зменшеної вартості, визначається за формулою:

$$\text{На} = \text{2/т} \cdot 100\% \tag{3.7}$$

де *Т* –строк корисного використання ПК, дорівнює 5 років.

Розрахуємо річну норму амортизації за формулою (3.7):

$$Ha = 2/5 \cdot 100\% = 40\% = 0{,}40 \text{ [частки одиниці]}.$$

Строк корисного використання ліцензійного програмування дорівнює 5 років.

Річна норма амортизації на ліцензійне програмне забезпечення визначається за формулою (3.7):

$$Напз = 2/5 \cdot 100\% = 40\% = 0{,}40 \text{ [частки одиниці]}.$$

Ліцензійне програмне забезпечення, яке використовується в даному випадку Microsoft Windows 7 Professional. Його вартість 8000 грн.

Вартість 1 години машинного часу ПК визначаються за формулою (3.5):

$$Смч = 1{,}1 \cdot 247 \cdot 1{,}60 + \frac{17500 * 0{,}40}{1920} + \frac{8200 * 0{,}40}{1920} = 440{,}08 \text{ [грн/год]}$$

Розрахуємо вартість машинного часу за формулою (3.4):

$$З_{мч}=(70+20+20+32)\cdot 440{,}08 = 62491{,}36 \text{ [грн]}.$$

Отже, підставивши отримані результати у формулу (3.2), отримаємо величину витрат на розробку моделей:

$$К_{пз} = 19760+62491{,}36=82251{,}36 \text{ [грн]}.$$

3.1.3 Розрахунок капітальних витрат

Загальні капітальні витрати на розробку визначаються за формулою:

$$КЗ = Кпз + Кнавч + Кн \text{ [грн]}, \quad (3.8)$$

де *Кнавч* - витрати на навчання технічних фахівців і обслуговуючого персоналу;

Кн - Витрати на встановлення обладнання та налагодження системи.

Дані о витратах на розробку моделей зводимо в таблицю 3.3

Таблиця 3.3 – Витрати на розробку моделей

Кпз, грн	*Кнавч,* грн	*Кн,* грн
82251,36	5200	1100

Отже, капітальні витрати становлять:

$$КЗ = 82251,36+5200+1100=88551,36 \text{ [грн]}.$$

3.2 Висновки

В економічному розділі було розраховано:

1 Трудомісткість розробки імітаційної моделей кодекыв – 247 год;

2 Заробітна платня проектувальника – 19760 ,00 грн;

3 Витрати на розробку моделей – 82251,36 грн;

4 Капітальні витрати на розробку моделей кодеків 88551,36 грн.

ВИСНОВКИ

1 Виконано аналітичний огляд методів побудови різних кодеків.

2 Виконано класифікація коригувальних кодів.

3 Розроблені імітаційні моделі найбільш широко застосовуваних кодеків.

4 Результати дослідження характеристик імітаційних моделей кодеків показали:

1) Кодеки, що використовують блокові лінійні коди (Хеммінга, БЧХ (Боуза-Чоудхурі-Хоквенгема), Ріда-Соломона) мають практично однакову ефективність з точки зору виправлення помилок. Число и типи помилок, Які могут буті виправлені, залежаний тільки від характеристик коду.

Коди Ріда-Соломона особливо добре підходять для коригування кластерів помилок (коли невірнімі віявляються великі групи біт кодового слова, які йдут поспіль).

2) Циклічний надлишковий код (CRC-код) - це дуже потужний і широко використовуваний метод виявлення помилок передачі інформації. Він забезпечує виявлення помилок з високою ймовірністю. Головна особливість значення CRC полягає в тому, що він однозначно ідентифікує

вихідну бітову послідовність і тому використовується в різних протоколах зв'язку, а також для перевірки цілісності блоків даних.

3) На відміну від блокових, згорткові коди мають наступні переваги:

- згорткові коди дозволяють виробляти кодування і декодування потоків даних безперервно в часі;

- згорткові коди не потребують блокової синхронізації;

- застосування згорткових кодів дозволяє досягти дуже високої надійності інформації, що передається.

ПЕРЕЛІК ПОСИЛАНЬ

1 Зюко А.Г., Фалько А.И., Панфилов И.П., Банкет В.Л., Иващенко П.В. Помехоустойчивость и эффективность систем передачи информации. М.:Радио и связь. 1985.

2 Методы повышения энергетической и спектральной эффективности цифровой радиосвязи: учеб. пособие / В. А. Варгаузин, И. А. Цикин. — СПб.: БХВ-Петербург, 2013. — 352 с.

3. Голиков А.М., Уваровский В.Д. Исследование многоуровневых методов модуляции сигналов, используемых в космических системах связи, на базе аппаратуры и ПО labVIEW 2010. Методические указания по лабораторным работам – Томск: Том. гос. ун-т систем управления и радиоэлектроники, 2011. – 50 с.

4 Банкет В.Л. Помехоустойчивое кодирование в телекоммуникационных системах: учебн. пособие. - Одесса: ОНАС им А.С. Попова, 2011. - 104 с.

3 Васильев А.В. Технико-экономическое обоснование дипломных проектов (работ): Учеб. пособие/ Изд-во СПбГЭТУ, 2002. - 24 с.

4 Экономика связи: Учебник для вузов/О. С. Срапионов, М. А. Горелик, В. И. Холодарь и др.; Под ред. О. С. Срапионов. – М.: Радио и связь, 2012. –

№	Формат	Найменування	Кількість листів	Примітки

320 с.

5 Грузинов В.П., Грибов В.Д. Экономика предприятия: Учеб. пособие – М.: Финансы и статистика, 2005. – 208 с.

6 Экономика предприятия / Под ред. Е.Л.Кантора. – СПб.: Питер, 2006. – 352 с.

7 Ворона, В. А. Радиопередающие устройства. Основы теории и расчета :

учеб. пособие для вузов / В. А. Ворона. – М. : Горячая линия-Телеком, 2007. –

384 с.

ДОДАТОК А. Відомість матеріалів дипломного проекту

Документація				
1	А4	Реферат		
2	А4	Список умовних скорочень		
3	А4	Зміст		
4	А4	Вступ		
5	А4	Стан питання. Постановка задачі		
6	А4	Спеціальна частина		
7	А4	Економічний розділ		
8	А4	Висновки		
9	А4	Перелік посилань		
10	А4	Додаток А		
11	А4	Додаток Б		
12	А4	Додаток В		
12		Матеріали дипломного проекту на оптичному носії		Оптичний диск

ДОДАТОК Б. Відгук керівника економічного розділу

Керівник розділу

к.е.н., доцент _____ Романюк Н.М.

(підпис) (прізвище, ініціали)

ДОДАТОК В Відгук керівника дипломної роботи

ВІДГУК

на магістерську дипломну роботу

Студента(ки) _____ гр.

(прізвище, ім'я)

на тему: _____

_____

_____

Актуальність теми _____

Повнота розкриття теми

Теоретичний рівень

Практична значущість

Самостійність виконання роботи

Якість оформлення, загальна та спеціальна грамотність

Переваги та недоліки роботи

Загальна оцінка роботи та висновок щодо рекомендації до захисту в ДЕК

Науковий керівник

к.ф.-м.н., професор Гусєв О.Ю.

(посада) (підпис) (ініціали, прізвище)

«____»_____2020 р.