П.Ф. ОЛЕКСЕНКО
В.В. КОВАЛЬ
Г.М. РОЗОРІНОВ
Г.О. СУКАЧ

ТЕОРЕТИЧНІ ОСНОВИ ЗАВАДОСТІЙКОГО КОДУВАННЯ

НАУКОВА ДУМКА

ПІДРУЧНИК

НАЦІОНАЛЬНА АКАДЕМІЯ НАУК УКРАЇНИ
ІНСТИТУТ ФІЗИКИ НАПІВПРОВІДНИКІВ ім. В.Є. ЛАШКАРЬОВА
МІНІСТЕРСТВО ОСВІТИ І НАУКИ, МОЛОДІ ТА СПОРТУ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ "КПІ"
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ БІОРЕСУРСІВ І ПРИРОДОКОРИСТУВАННЯ УКРАЇНИ

П.Ф. ОЛЕКСЕНКО, В.В. КОВАЛЬ, Г.М. РОЗОРІНОВ, Г.О. СУКАЧ

ТЕОРЕТИЧНІ ОСНОВИ ЗАВАДОСТІЙКОГО КОДУВАННЯ

ЧАСТИНА II

Підручник для вищих навчальних закладів

За редакцією академіка НАН України В.Ф. МАЧУЛІНА

КИЇВ НАУКОВА ДУМКА 2012

Затверджено Міністерством освіти і науки,
молоді та спорту України
як підручник для студентів
телекомунікаційних спеціальностей
вищих навчальних закладів
(Лист № 1/11-2775 від 2 квітня 2010 року)

Розглянуто методи створення, а також практичні схеми декодування циклічних кодів з використанням різних принципів декодування. Особливу увагу приділено аналізу методів кодування й декодування ефективних і зручних у реалізації недвійкових циклічних кодів Ріда—Соломона, які дають змогу виявляти і виправляти помилки в блоках даних, тобто вони є ефективними в каналах з пам'яттю. Розглянуто алгоритм Берлекемпа—Мессі розв'язку ключового рівняння. Описано методи виправлення помилок і стирань. Наведено основні принципи реалізації кодера і декодера Ріда—Соломона.

Для студентів вищих навчальних закладів, які повинні вміти не тільки побудувати апаратуру кодування-декодування, а й включити цю апаратуру в інформаційно-комунікаційну систему.

Р е ц е н з е н т и:
В.П. БОЮН, доктор технічних наук, професор,
член-кореспондент НАН України
Р.В. КОНАКОВА, доктор технічних наук, професор
Ю.Г. САВЧЕНКО, доктор технічних наук, професор

Видання здійснено за державним замовленням
на випуск видавничої продукції

Науково-видавничий відділ фізико-математичної та технічної літератури

Редактор *О.А. Микитенко*

ISBN 978-966-00-1139-7

ПЕРЕДМОВА

Ця праця є логічним продовженням підручника "Теоретичні основи завадостійкого кодування" (ч. 1). У ній проаналізовано методи створення циклічних кодів. Наведено основні методи декодування циклічних кодів як лінійних кодів. Висвітлено питання, пов'язані з теорією і практикою застосування цих та подібних кодів. Розглянуто методи декодування циклічних кодів за дистанційним принципом у моделі відстані одиничної, подвійної та потрійної помилок, а також декодування за моделлю відстані помилок кодів БЧХ, що широко використовуються на практиці.

Особливу увагу приділено аналізу методів кодування і декодування ефективних і зручних у реалізації недвійкових циклічних кодів Ріда—Соломона (РС-кодів), що були винайдені в 1960 р. співробітниками лабораторії Лінкольна Массачусетського технологічного інституту Ірвіном Рідом і Густавом Соломоном. Поява їх стала видатною подією в теорії та техніці завадостійкого кодування. Уперше код Ріда—Соломона набув ужитку в 1982 р. під час серійного випуску компакт-дисків. Сфера їх застосування дуже широка: кодери/декодери Ріда—Соломона можна знайти і в стрічкових запам'ятовувальних пристроях, і в контролерах оперативної пам'яті, і в модемах, і в жорстких дисках, і в CD-ROM/DVD приводах тощо. Код Ріда—Соломона, що виправляє t помилок, вимагає $2t$ перевірних символів і за його допомогою виправляються довільні пакети довжиною t і менше. Елементами кодового вектора є не біти, а групи бітів (блоки). Вони дають змогу виправляти помилки в блоках даних. Значного поширення набули РС-коди, які працюють з байтами (октетами).

Способи кодування і декодування РС-кодів досить добре розроблені з погляду теорії та їх реалізації. Вони базуються на процедурах виправлення стирань, помилок, а також сумісного виправлення помилок і стирань. У цьому аспекті розглянуто ефективну процедуру декодування з виправленням помилок за алгоритмом Форні.

У праці розглядається алгоритм Берлекемпа—Мессі розв'язку ключового рівняння, запропонований в 1969 р. Елвіном Берлекемпом і Джеймсом Мессі. Цей алгоритм побудовано з використанням ітераційної процедури: за кожної ітерації має зберігатися як многочлен зв'язків, так і додаток. Для кожного нового члена пе-

редбачається перевірка правильності прогнозу цього символу поточним многочленом зв'язку. Якщо прогноз правильний, то многочлен зв'язків не змінюється, а додаток множиться на x, а якщо неправильний, то змінюють поточний многочлен зв'язків, додаючи до нього додаток. Потім перевіряють, чи збільшилася довжина регістра: якщо ні, то наявний додаток залишають; якщо так, то кращим додатком вважають попередній многочлен зв'язків. У разі недвійкових полів додаток нормують так, щоб незв'язність дорівнювала "1". Далі за кожного виправлення цей нормалізований додаток множать на значення наявної незв'язності.

Процедуру об'єднання переваг різних способів кодування реалізовано через застосування каскадного кодування. При цьому інформація спочатку кодується одним кодом, а потім іншим, як наслідок одержується код-добуток. Наприклад, популярною є наступна конструкція: дані кодуються кодом Ріда—Соломона, потім чергуються (при цьому символи, що розташовані близько, поміщаються далеко один від іншого) і кодуються згортковим кодом. На приймачі спочатку декодується згортковий код, далі здійснюється зворотне чергування (при цьому пачки помилок на виході згорткового декодера потрапляють у різні кодові слова РС-коду), потім здійснюється декодування коду Ріда—Соломона.

У праці висвітлено питання кодування та декодування згорткових кодів, які на відміну від блокових, не ділять інформацію на фрагменти і працюють з нею як з суцільним потоком даних, тобто з потоками бітів або символів довільної протяжності. Згорткові коди, як правило, породжуються дискретною лінійною інваріантною у часі системою. Тому, на відміну від більшості блокових кодів, згорткове кодування дуже проста операція, що не можна стверджувати про декодування. Кодування загортковим кодом виконується за допомогою регістра зсуву, сигнали з відводів від якого підсумовуються за модулем 2. Таких сум може бути дві або більше. Декодування згорткових кодів, як правило, виконується за алгоритмом Вітербі, який намагається відновити передану послідовність згідно з критерієм максимальної правдоподібності. Згорткові коди ефективно працюють у каналі з білим шумом, але погано справляються з пакетами помилок. Крім того, якщо декодер помиляється, то на його виході завжди виникає пакет помилок.

Взаємозв'язок теорії та практики завадостійкого кодування — це основа сучасних інформаційно-комунікаційних систем. Таке кодування обумовлює науково-технічний прогрес. Тому тут наша ціль — висвітлити питання, пов'язані зі зближенням розрахунково-аналітичного апарату теорії та інженерної практики застосування завадостійких кодів.

ДЕКОДУВАННЯ ЦИКЛІЧНИХ КОДІВ ЯК ЛІНІЙНИХ КОДІВ

16. 1. ЦИКЛІЧНІ КОДИ ЯК ПІДКЛАС ЛІНІЙНИХ КОДІВ

Зазвичай усі блокові коди можна розділити на дві нерівномірні групи: більша із них — роздільні лінійні коди, менша — роздільні нелінійні коди. Як зазначалося (див. ч. I), лінійні коди утворюють векторний простір. Як правило, вектор двійкового коду задається у вигляді многочлена (а не комбінації 0,1).

Лінійним кодам притаманна така важлива властивість: два кодові слова можна додати, використовуючи відповідне визначення суми, і отримати третє кодове слово. У разі звичайних двійкових кодів ця операція є порозрядним додаванням двох кодових слів за модулем 2 (тобто $1 + 1 = 0$, $1 + 0 = 1$, $0 + 0 = 0$). Наведена властивість має два важливих наслідки. Перший з них полягає в тому, що лінійність істотно спрощує процедури кодування і декодування, даючи змогу подати кожне кодове слово у вигляді лінійної комбінації невеликого числа виділених кодових слів, так званих базисних векторів. Другий наслідок полягає в тому, що лінійність істотно спрощує завдання обчислення параметрів коду, оскільки відстань між двома кодовими словами у цьому разі еквівалентна відстані між кодовим словом, що складається цілком з нулів, і деяким іншим кодовим словом. Лінійність коду спрощує також його побудову і реалізацію. За великої довжини практично можуть бути використані тільки лінійні коди. Майже всі схеми кодування, що на сьогодні використовуються на практиці, ґрунтуються на лінійних кодах.

Таким чином, при обчисленні параметрів лінійного коду досить з'ясувати, що відбувається у разі передавання кодового слова, яке складається тільки з нулів. Обчислення параметрів спрощується ще і тому, що відстань Хеммінга між даним кодовим словом і нульовим кодовим словом дорівнює числу ненульових елементів даного кодового слова. Таке число часто називають вагою Хеммінга цього слова, і список, що містить число кодових слів кожної ваги, можна використовувати для обчислення характеристик коду за допомогою адитивної межі. Такий список називають спектром коду.

Незважаючи на те, що декодування лінійних кодів значно простіше, ніж декодування більшості нелінійних, для більшості кодів цей процес досить складний. З теорії кодування відомо, що циклічні коди можна подати як лінійні — вони є підкласом лінійних кодів. Можна також вважати, що циклічні коди — це специфічне подання лінійних кодів, тобто при декодуванні циклічних кодів можуть безпосередньо використовуватися методи декодування лінійних кодів. Тому далі розглядається, яким чином можна ефективно використовувати ці методи.

16.2. ЦИКЛІЧНИЙ НАДЛИШКОВИЙ КОД

Циклічний надмірний код (англ. Cyclic redundancy code, CRC) — спосіб цифрової ідентифікації деякої послідовності даних, який полягає в обчисленні контрольного значення її циклічного надмірного коду. Під надмірністю розуміють використання більшого числа символів, ніж це мінімально необхідно. Значною надмірністю володіє мова. Виникає питання, як побудувати код, щоб надмірність була мінімальною. Основна вимога до побудови не надто надмірних (ефективних) кодів — чим більша вірогідність появи повідомлення, тим меншою повинна бути його довжина.

Суттєва властивість циклічного коду полягає в наступному. Якщо деякий вектор належить циклічному коду, то будь-який інший вектор, отриманий з даного шляхом довільного числа циклічних зрушень, також належить циклічному коду. Ідея побудови циклічного коду базується на понятті незвідного многочлена, який ділиться тільки сам на себе і на одиницю, тобто не може бути розкладений на простіші многочлени. Сам же незвідний многочлен є дільником многочлена $(x^n + 1)$ без залишку. Незвідні многочлени в теорії циклічних кодів відіграють роль породжувальних многочленів або поліномів. Ці поліноми табульовані.

Зі всієї сукупності методів контролю з використанням циклічної структури групового сигналу найбільшого поширення набув метод контролю первинних цифрових трактів CRC-4 (число 4 вказує на степінь многочлена вхідних даних та на степінь ієрархії цифрових трактів).

У разі передавання потоку даних по мережі зв'язку виникає необхідність перевірки якісних показників первинного цифрового каналу від його вхідних блоків до вихідних.

З 256 бітів, що створюють стандартний цикл 2-мегабітного сигналу, сигнали з 1-го по 31-й канальний інтервал (КІ) є випадковими. Тому зі всього групового сигналу можна ідентифікувати тільки 7 біт циклового синхросигналу, що дає змогу виявляти бітові помилки без зупинки зв'язку. Проте такі 7 біт становлять лише 1,4 % загального обсягу інформації, що передається. Проблема забезпечення контролю помилок в інших 31 каналу оптимально вирішується застосуванням методу контролю з використанням надмірності циклічного сигналу CRC-4. Цей метод визначено в 1980-х роках рекомендацією МСЕ-Т, але великого поширення набув тільки останнім часом через труднощі реалізації схемотехніки, яку можна здійснити тільки методами мікросхемотехніки.

Суть методу полягає в тому, що цифровий сигнал розбивається на групи, які отримали назву блоків або субнадциклів. На кінці передавального тракту підраховується сума символів блока. Ця інформація у складі групового сигналу передається на приймальний кінець тракту, де підрахунок суми символів повторюється, і результати підрахунку на вихідному і приймальному трактах порівнюються. Збіг результатів інтерпретується як відсутність помилок при передачі блока. Розбіжність результатів вказує на наявність помилок при передачі даного блока. Для передачі результатів порівняння у зворотному напрямі, на кінець передавального тракту, використовуються вільні біти службових канальних інтервалів групового сигналу зворотного напряму.

16 циклів, що ідуть один за одним, утворюють надцикл, який, у свою чергу, поділяється на два субнадцикли (1-й та 2-й) по 8 циклів кожен. Таким чином, часовий інтервал CRC-4 визначається так: 125 мкс × 16 = 2 мс.

Для формування сигналу CRC-4 сума бінарних символів кожного субнадциклу ділиться на поліном четвертого степеня ($x^4 + x + 1$). Результат ділення, що становить 4 бінарних символи, вводиться в груповий сигнал у позиціях від С1 до С4. Приймальна сторона використовує аналогічний метод, щоб потім порівняти кодове слово, що надходить від передавача, з результатом, отриманим на приймальному кінці. Якщо вказані слова різні, то субнадцикл, що дорівнює 2048 бітам, переданий з помилками.

Перевага цього методу — за його допомогою можна контролювати цифровий потік без зупинки зв'язку і незалежно від його складу. Водночас у разі його використання неможливо точно вказати, які біти з тисяч переданих були прийняті з помилками.

Надцикловий сигнал CRC-4 застосовується для забезпечення синхронізації по бітах від С1 до С4. Біти Е (Е1 і Е2 для 1-го і 2-го субнадциклів) інвертуються для збереження структури надциклу в моменти виявлення помилок. Таким чином, приймальна сторона може інформувати сторону, що передає, про виявлення помилок передачі. Після того, як встановиться циклова синхронізація, сигнали CRC-4 передаватимуться неперервно. Втрата синхронізації CRC-4 відбувається тоді, коли більше ніж 914 сигналів CRC-4, що передаються протягом 1 с, не відповідатимуть нормованим.

Аналогічно відбувається перевірка цифрових трактів інших ступенів ієрархії. Змінюється лише розмір блоків і степінь полінома: 6-й для CRC-6, що використовується для контролю ІКМ-120, 8-й для CRC-8 — для контролю ІКМ-480.

16.3. ФОРМАЛІЗОВАНИЙ АЛГОРИТМ РОЗРАХУНКУ ЦИКЛІЧНОГО НАДЛИШКОВОГО КОДУ

Щоб отримати контрольну суму, необхідно згенерувати той або інший поліном. Основна вимога до полінома — його степінь повинен дорівнювати довжині контрольної суми в бітах. При цьому старший біт полінома обов'язково повинен дорівнювати "1".

Розглянемо таку процедуру. З файла береться перше слово. Завантаження починається з молодшого розряду, як правило, зсувом слова на один розряд вліво (множення). Після зсуву (множення) втрачається старий старший біт, а молодший біт звільняється (обнуляється). На місце молодшого біта завантажується черговий біт з файла. Операція повторюється доти, поки не завантажиться останній біт файла. Після проходження всього файла в слові залишається залишок, який і є контрольною сумою.

Отже, алгоритм розрахунку циклічного надмірного коду базується на властивостях ділення із залишком двійкових многочленів, тобто многочленів над кінцевим полем $GF(2)$. Значення CRC по суті — це залишок від ділення многочлена, що відповідає вхідним даним, на якийсь фіксований *породжувальний многочлен*.

Кожній кінцевій послідовності бітів $a_0, a_1,...,a_{N-1}$ взаємно однозначно ставиться у відповідність двійковий многочлен $\sum_{n=0}^{N=1} a_n x^n$, послідовність коефіцієнтів якого є початковою послідовністю. Наприклад, послідовність бітів 1011010 відповідає многочлену

$$P(x) = 1 \cdot x^6 + 0 \cdot x^5 + 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 0 \cdot x^0 = x^6 + x^4 + x^3 + x^1.$$

Кількість різних многочленів степеня меншого ніж N дорівнює $2N$, що збігається з числом усіх двійкових послідовностей довжиною N.

Значення CRC з породжувальним многочленом $G(x)$ степеня N визначається як бітова послідовність довжиною N, що є многочленом $R(x)$, який отримано в залишку при діленні многочлена $P(x)$, що є вхідним потоком бітів, на многочлен $G(x)$, тобто

$$R(x) = P(x) \cdot x^N \bmod G(x),$$

де $R(x)$ — многочлен, що є значеннями CRC; $P(x)$ — многочлен, коефіцієнти якого є вхідними даними; $G(x)$ — породжувальний многочлен; N — степінь породжувального многочлена.

Множення x^N здійснюється приписуванням N нульових бітів до вхідної послідовності, що покращує якість хешування (високу перемішуваність та швидкий алгоритм обчислення) для коротких вхідних послідовностей.

При діленні з залишком степінь многочлена-залишку строго менший, ніж степінь многочлена-дільника, тобто при діленні на многочлен $G(x)$ степеня N можна отримати $2N$ різних залишків від ділення. При "правильному" виборі породжувального многочлена $G(x)$ залишки від ділення на нього матимуть потрібні властивості хешування. Швидкість забезпечується тим, що степінь породжувального многочлена зазвичай пропорційний довжині байта або машинного слова (наприклад, 8, 16 чи 32).

16.4. ЗАГАЛЬНОПРИЙНЯТІ МЕТОДИ КОДУВАННЯ ЦИКЛІЧНИХ КОДІВ

Відомо три методи кодування і відповідно три способи побудови кодувальних пристроїв, перші два з яких засновані на властивостях породжувального многочлена.

Модель 1. Згідно з правилом

$$\beta(x) = v(x)g(x),$$

комбінацію циклічного коду $\beta(x)$ можна отримати при множенні комбінації простого коду $v(x)$ на породжувальний многочлен $g(x)$. За цим правилом і побудовано кодувальний пристрій, тобто він є пристроєм множення.

Такий пристрій простий в реалізації, але має недолік: отримуваний циклічний код не є систематичним. Прийнята кодова комбінація в явному вигляді не містить інформаційних символів. Щоб виділити інформаційні символи потрібно додатково, після прийому, виконати операцію ділення: $\beta(x)/g(x) = v(x)$, що ускладнює структуру приймача.

Модель 2. Дана модель передбачає обчислення залишку $r(x)$ від ділення інформаційного многочлена $v(x)$, заздалегідь помноженого на x^k, на породжувальний

многочлен, тобто

$$v(x)x^k = g(x)g'(x) \oplus r(x).$$

Розглянемо многочлен $\beta(x) = g(x)g'(x) = v(x)x^k \oplus r(x) = v(x)x^k \oplus r(x)$ за модулем $(x^n + 1)$. Коефіцієнти $n–k$ біля старших розрядів розглядаються як m інформаційних символів, а біля молодших розташовані перевірні символи. Отже, отриманий код буде систематичним.

Будуючи кодер, використовують одночасно операції множення на $v(x) = x^k$ і ділення на $g(x)$.

Модель 3. Дана модель ґрунтується на властивостях перевірного многочлена. Відомо, що

$$\beta(x)h(x) \equiv 0 \text{ за mod } (x^n + 1).$$

Нехай $(\beta H^{(i)})$, $i = 1, 2, ..., n–1$, — скалярний добуток, де $H^{(i)}$ — вектор, коефіцієнти якого зрушені на i розрядів у бік старшого розряду (вліво). Вектор β ортогональний вектору H, записаному в зворотному порядку, і всім векторам, утвореним циклічним зрушенням.

Якщо інформаційні символи розміщені на перших m позиціях кодового вектора $\beta(x)$, то за допомогою перших k рівнянь наведеної вище системи можна визначити контрольні символи.

Зауважимо, що циклічні коди спрощують процес кодування-декодування, особливо у разі виявлення помилок. Крім того, циклічні коди — це лінійні коди, яким притаманна така властивість: якщо многочлен — кодове слово, то його циклічна перестановка — також кодове слово.

Циклічні коди використовуються при послідовному зв'язку і застосовуються не лише для кодового захисту, а й для перетворення інформаційного слова з метою виключення довгих послідовностей одиниць. Довгі послідовності нулів або одиниць погано передаються по лініях зв'язку.

Для кожного циклічного коду потрібне початкове число або початковий многочлен (обов'язково простий). Перетворення здійснюється таким чином:

а) до інформаційного слова додається число нулів, що дорівнює кількості розрядів початкового числа (тобто многочлен множиться на $2k$, де k — порядок початкового многочлена або число розрядів початкового числа);

б) інформаційне слово ділиться на початковий многочлен;

в) до отриманого результату дописується справа залишок від ділення, слово з кодовим захистом матиме $(n + k)$ розрядів;

г) при зворотному перетворенні за відсутності збоїв зайві нулі в кінці слова відкидаються;

д) за наявності збоїв у кінці слова замість нулів буде фігурувати синдром помилки, може також з'явитися зменшення або збільшення числа розрядів інформаційного слова. Керуючись ознаками помилки, контролер відновлює передане кодове повідомлення.

Циклічні коди побудовані на базі бітового рядка як рядки коефіцієнтів полінома. Рядок k-бітової інформації відповідає многочлену k – 1-го степеня. Лівіший біт рядка — це коефіцієнт біля старшого розряду. Наприклад, рядок 110001 зображує многочлен $x^5 + x^4 + x^0$. Коефіцієнти многочлена належать полю $GF(2)$ — полю за модулем 2. Слова циклічного коду зручно подавати у вигляді незвідних многочленів, тобто поліномів, які можуть бути запи-

сані у вигляді добутків многочленів нижчих степенів. Загалом, якщо не вказане інше, то вважається, що циклічний код є *двійковим*, тобто може набувати значення "0" чи "1".

Основна ідея передавання корисної інформації, її кодування і декодування за допомогою циклічних кодів полягає в тому, щоб пересилати тільки ті повідомлення, многочлени яких діляться на деякий фіксований многочлен $G(x)$. Якщо ми отримуємо повідомлення, у якому многочлен не ділиться на $G(x)$, то це означає, що при передачі сигналу він був спотворений. Ми не помітимо помилок, якщо вони один допустимий многочлен (тобто многочлен, що ділиться на $G(x)$) перетворили на інший також допустимий. Многочлен $G(x)$ тим кращий для передачі інформації, чим більше середня відстань Хеммінга на парах допустимих поліномів.

Як зазначалося, у кодуванні за допомогою циклічних кодів використовується спосіб додавання до інформаційного слова деякої кількості додаткових бітів так, щоб результуючий многочлен ділився на $G(x)$. Відправник і одержувач наперед домовляються про конкретний поліном-генератор $G(x)$.

16.5. ВИЗНАЧЕННЯ КОНФІГУРАЦІЇ СИНДРОМУ

16.5.1. Загальні положення щодо виявлення і виправлення помилок у циклічних кодах

Помилки в прийнятій кодовій комбінації можна виявити і виправити за допомогою завадостійкого коду. Корегувальні можливості коду визначаються його надлишковістю. Коди реалізуються для таких режимів: 1) виявлення помилок; 2) виправлення помилок; 3) одночасне виявлення і виправлення помилок.

Будь-яке кодове слово циклічного коду ділиться на незвідний многочлен без залишку, тому ознакою наявності помилки в прийнятому слові є ненульовий залишок від ділення прийнятого слова на незвідний многочлен. Проте наявність ненульового залишку лише вказує на факт існування помилки, тобто помилки виявляються, але за виглядом залишку не можна з'ясувати місце її виникнення. Для корекції t і менше помилок використовують такий алгоритм: 1) прийняте слово ділиться на незвідний многочлен; 2) підраховується вага залишку; 3) якщо вага не перевищує корегувальну здатність коду (t), то залишок підсумовується з діленим. Отримана сума — є правильне слово. Якщо вага залишку більша ніж t, то прийняте слово циклічно зсувається вліво на один розряд і ділиться на $P(xr)$. Далі аналізується вага залишку. Якщо вона не більша ніж t, то залишок додається до діленого. Отримана сума циклічно зсувається вправо на один розряд. Результат цієї операції — скореговане слово. Якщо вага залишку більша ніж t, то ділене ще раз зсувається вліво.

16.5.2. Знаходження конфігурації синдрому з використання матриці перевірки на парність

При декодуванні лінійних кодів після прийому всієї кодової бітової послідовності визначається синдром за допомогою матриці перевірки на парність. У разі циклічного коду синдром можна отримати як залишок від ділен-

ня на породжувальний многочлен, який відображається на виходах регістра схеми ділення в той момент, коли завершується введення всієї кодової бітової послідовності, що приймається. Тобто виявляється, що у разі циклічних кодів до моменту завершення введення кодової послідовності, що приймається, синдром є вже відомою величиною.

У разі циклічних кодів значення на виходах регістра показують залишок від ділення, який збігається з синдромом, тому конфігурацію значень відповідних виходів регістра зручно назвати конфігурацією синдрому, а саме:

Лінійний код	*Циклічний код*
Синдром	Конфігурація синдрому (конфігурація значень виходів регістра схеми ділення, на якій отримано залишок)

тобто виявляється, що синдром, властивий лінійним кодам, можна отримати у вигляді конфігурації синдрому на виходах регістра схеми ділення.

У разі, коли декодування циклічного коду здійснюється за принципами лінійного коду, знаходження синдрому відбувається в три етапи.

1. Визначення конфігурації синдрому.
2. Пошук конфігурації помилок, яка відповідає конфігурації синдрому.
3. Виправлення помилок шляхом підсумовування за модулем 2 конфігурації помилок з прийнятою кодовою бітовою послідовністю.

Щодо першого етапу — знаходження конфігурації синдрому, то його можна отримати у вигляді сигналів на виходах регістра схеми ділення на породжувальний многочлен на той момент часу, коли закінчується введення кодової бітової послідовності, що приймається.

Пристрій ділення вхідного сигналу на породжувальний многочлен зображено на рис. 1. При роботі пристрою протягом перших k тактів відбувається заповнення елементів пам'яті. Отже, до k-го такту зворотний зв'язок не працює, і в схемі здійснюється звичайний зсув. Починаючи з $(k + 1)$-го такту на виході пристрою починають з'являтися елементи частки від ділення, і вступає в дію зворотний зв'язок. Після l (l — старший ступінь вхідного сигналу) тактів на виході пристрою буде сформовано останній елемент частки від ділення, а на виході регістрів буде записано залишок від ділення — синдром помилки.

Це означає, що у разі коду БЧХ (n, k) порядок породжувального многочлена $m = n - k$ дорівнюватиме числу перевірних бітів, тому можна отримати конфігурацію синдрому у вигляді вихідних сигналів регістра, що складається з m бітів (рис. 1).

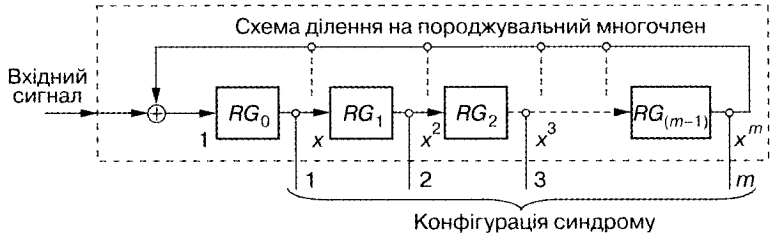


Рис. 1. Схема знаходження конфігурації синдрому

16.6. ВИЗНАЧЕННЯ КОНФІГУРАЦІЇ ПОМИЛОК

Для використання на другому етапі знаходження конфігурації помилок, які відповідають конфігурації синдрому, мабуть, найбільш простим та швидкодійним пристроєм (і методом) є структурна схема, базована на постійному запам'ятовувальному пристрої (ПЗП). Якщо конфігурацію синдрому прийняти як вхідну адресу ПЗП, а в тіло останнього за відповідною адресою записати конфігурацію помилок, то на виході ПЗП можна безпосередньо отримувати конфігурацію помилок, яка відповідає конфігурації синдрому (рис. 2).

Наведений на рис. 2 вихідний ідентифікуючий сигнал *ID* слугує для того, щоб у разі використання як адреси ПЗП конфігурації синдрому для визначення конфігурації помилок, яка перевершує можливості виправлення помилок, мати можливість показати, що помилка не підлягає виправленню. Так, якщо прийняти, що для помилки, яка підлягає виправленню, $ID = 0$, а для помилки, яка не підлягає виправленню, $ID = 1$, то за допомогою цього ідентифікатора можна визначати, придатною чи ні для використання є прийнята кодова бітова послідовність після виправлення помилок.

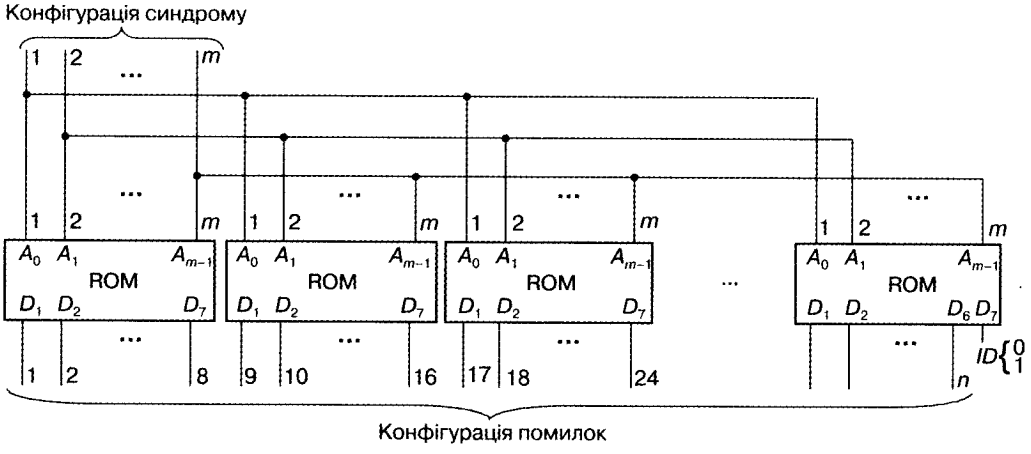


Рис. 2. Схема визначення конфігурації помилок за допомогою ПЗП

16.7. ПРАКТИЧНІ СХЕМИ ДЕКОДУВАННЯ

Перед третім етапом, тобто додаванням конфігурації помилок до кодової бітової послідовності за модулем 2, що приймається, необхідно, перш за все, сформувати кодову бітову послідовність, що приймається. Цю послідовність можна сформувати за допомогою приймального буферного регістра. Іншими словами, для додавання кодової бітової послідовності з конфігурацією помилок між виходами приймального буферного регістра необхідно встановити суматори за модулем 2 (рис. 3).

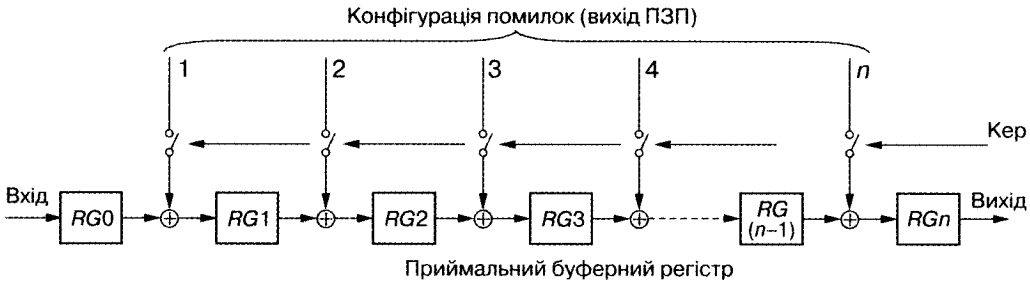


Рис. 3. Схема додавання кодової бітової послідовності, що приймається, з конфігурацією помилок

Структура схеми додавання передбачає, що якщо на виході ПЗП сформована "1", то між двома сусідніми регістрами вмикається суматор за модулем 2, а якщо сформовано "0", то вихід попереднього регістра безпосередньо підключається до входу наступного. Операція додавання конфігурації помилок і тієї кодової бітової послідовності, що приймається, яка знаходиться в приймальному буферному регістрі, повинна виконуватися тільки одноразово (див. рис. 3), тому використовується спеціальний сигнал керування **Кер**. У цьому разі сума конфігурації помилок та кодової бітової послідовності, що приймається, знаходиться за синхроімпульсом, який формується відразу після того, як на вихід надійде конфігурація помилок, яка відповідає конфігурації синдрому. За допомогою цього синхроімпульсу здійснюється виправлення помилок, і це означає, що надалі виправлені прийняті кодові бітові послідовності надходять на вихід пристрою. Як наслідок узагальнена схема декодування лінійного коду БЧХ (n, k) набуває вигляду, відповідного зображеному на рис. 4.

Подану схему декодування можна ще більш спростити, розробивши таблицю відповідності конфігурацій синдрому і помилок.

Наприклад, спробуємо декодувати код БЧХ (7, 4). Для цього необхідно, перш за все, побудувати таблицю відповідності конфігурації синдрому і конфігурації помилок. Така відповідність конфігурації синдрому і конфігурації помилок для цього коду наведена в табл. 1.

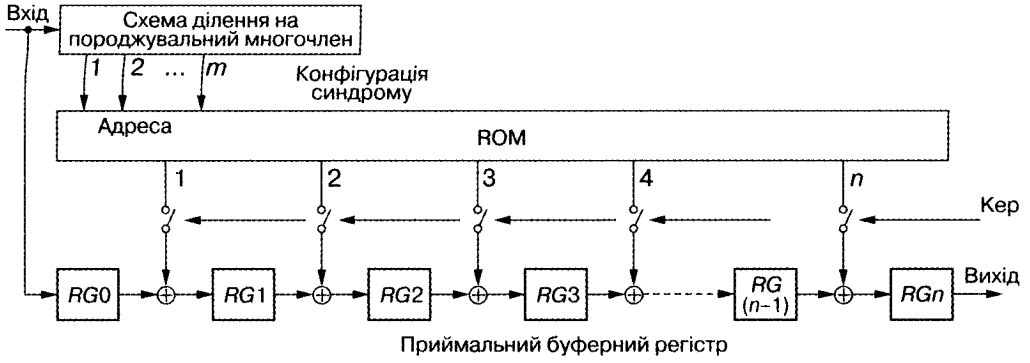


Рис. 4. Схема декодування лінійного коду БЧХ (n, k)

Таблиця 1. Відповідність конфігурацій синдрому і помилок для коду БЧХ (7, 4)

Конфігурація синдрому			Конфігурація помилок						
RG_0	RG_1	RG_2	e_0	e_1	e_2	e_3	e_4	e_5	e_6
Адреса ПЗП			Дані ПЗП						
A_0	A_1	A_2	D_0	D_1	D_2	D_3	D_4	D_5	D_6
0	0	0	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0
0	1	0	0	1	0	0	0	0	0
1	1	0	0	0	0	1	0	0	0
0	0	1	0	0	1	0	0	0	0
1	0	1	0	0	0	0	0	0	1
0	1	1	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	1	0

Для забезпечення відповідності з кодовою бітовою послідовністю, що приймається, конфігурацію помилок, яка записується в приймальному буферному регістрі, доцільно подавати зліва направо.

На підставі табл. 1 можна дійти висновку, що сумарна кількість конфігурацій одиничних помилок разом з конфігурацією відсутності помилок дорівнює восьми. Порядок породжувального многочлена ($x^3 + x + 1$) лінійного коду БЧХ (7, 4) дорівнює трьом, тому число виходів регістра схеми ділення на породжувальний многочлен також дорівнює трьом. Тому конфігурація синдрому, яка формується на виходах регістра, включає три біта. Число конфігурацій, яке можна сформувати з використанням трьох бітів, становить $2^3 = 8$. Отже, число конфігурацій помилок, які можуть бути виправлені, дорівнює числу конфігурацій, які можна подати конфігураціями синдрому, що дорівнює 8.

Таким чином, між числом конфігурацій помилок, які можуть бути виправлені, і числом конфігурацій синдрому, що складаються з трьох бітів, існує взаємно однозначна відповідність. Отже, для даного прикладу, число конфігурацій синдрому таке, що вони дають змогу виправляти тільки одиночні помилки, тому немає необхідності використовувати ідентифікуючий сигнал *ID*. Більш того, цей сигнал взагалі не можна використовувати.

На рис. 5 реалізовано схему декодування лінійного коду БЧХ (7, 4) і таблицю її функціонування. Адреси і вихідні дані ПЗП відповідають табл. 1.

Як ще один приклад розглянемо випадок коду БЧХ (15, 7). За своєю здатністю до виправлення помилок — це код з виправленням подвійних помилок, тому як конфігурації помилок необхідно розглядати всі конфігурації помилок, аж до конфігурацій з подвійними помилками. Перш за все слід визначити число конфігурацій помилок, які можуть бути виправлені.

Неважко побачити, що:

- число конфігурацій відсутності помилок дорівнює 1;
- число конфігурацій одиничних помилок дорівнює 15.

Тому загальне число таких конфігурацій дорівнює 16.

Число конфігурацій з подвійними помилками можна підрахувати наступним чином:

- число подвійних помилок разом із першим бітом і будь-яким іншим бітом дорівнює 14;
- число подвійних помилок, а також другий біт і будь-який інший біт (за винятком першого біта) дорівнює 13;
- число подвійних помилок разом із третім бітом і будь-яким іншим бітом (за винятком першого і другого бітів) дорівнює 12, і далі в напрямі зменшення подвійних помилок. Тобто з'ясовується, що число конфігурацій з подвійними помилками розраховується так:

$$14 + 13 + 12 + 11 + 10 + 9 + 8 + 7 + 6 + 5 + 4 + 3 + 2 + 1 = 105.$$

Отже, зрештою загальне число перерахованих вище конфігурацій становить:

(*число конфігурацій без помилок*) + (*число конфігурацій з одиничними помилками*) + (*число конфігурацій з подвійними помилками*) = 1 + 15 + 105 = 121.

Порядок породжувального многочлена коду БЧХ (15, 7) дорівнює восьми, тому конфігурація синдрому складається з восьми бітів. Крім того, число бітових комбінацій, які можна подати за допомогою восьми бітів конфігурації синдрому, становить $2^8 = 256$.

На підставі того, що

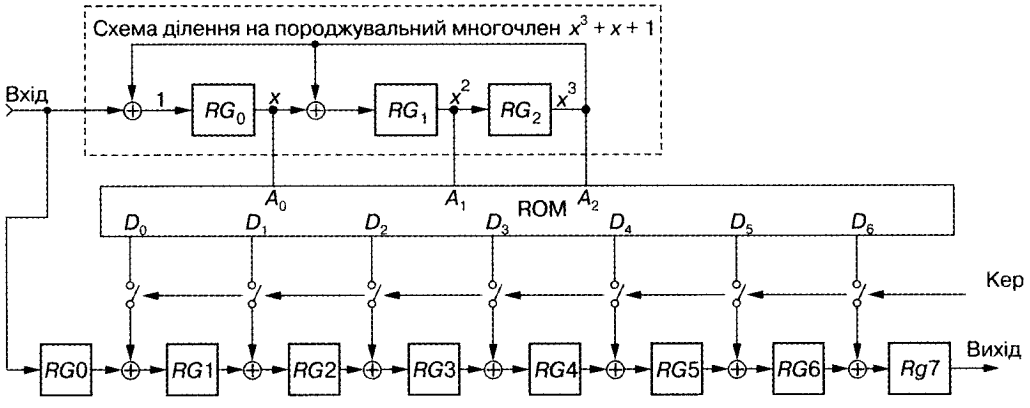
(*число конфігурацій синдрому, які можуть бути подані за допомогою восьми бітів*) – (*число конфігурацій помилок, які можуть бути виправлені*) = 256 – 121 = 135,

за допомогою цих бітів можна подати ще 135 станів, крім конфігурацій помилок, які можуть бути виправлені. Проте виявляється, що ці 135 конфігурацій помилок — це конфігурації, які виходять за рамки здатності коду БЧХ (15, 7) до виправлення помилок. Тому як ідентифікуючі сигнали ID ПЗП, що використовуються для декодування цього коду, можна застосувати, наприклад, такі *сигнали:*

- для 121 комбінації синдрому, що показують наявність помилок, які можуть бути виправлені, *ID* = 0;
- для 135 комбінацій синдрому, що показують наявність помилок, які не можуть бути виправлені, *ID* = 1.

Таблиця відповідності між конфігураціями синдрому і конфігураціями помилок, тобто таблиця відповідності між адресами ПЗП і даними, що зберігаються в пам'яті ПЗП, досить громіздка, оскільки число конфігурацій синдрому 256 дуже велике, тому вона не наводиться.

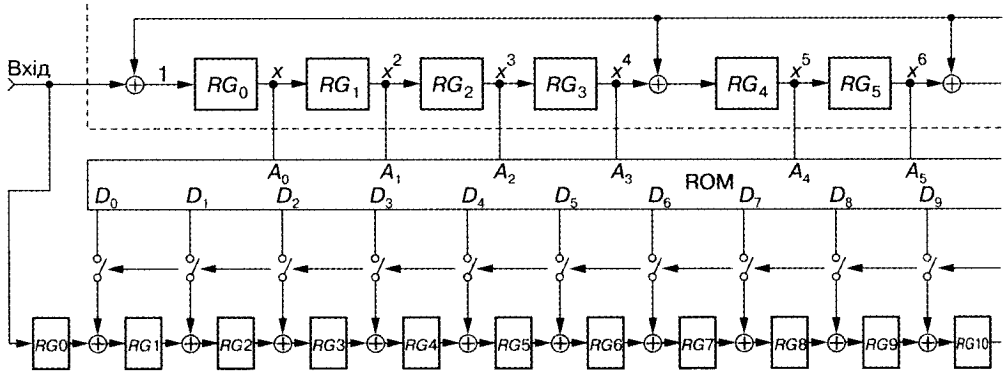
На рис. 6 показана схема декодування лінійного коду БЧХ (15, 7). Для даних, що відповідають конфігурації синдрому (адресі ПЗП), яка показує наявність комбінації помилок, що підлягають виправленню, з числом помилок, що не перевищує два, сигнал *ID* дорівнює нулю. Тоді як для даних, які відповідають конфігурації синдрому (адресі ПЗП) для більшого числа помилок, сигнал *ID* дорівнює одиниці.



Такт	Вхід	Конфігурація синдрому		
		RG_0	RG_1	RG_2
0	$a_3 + e_6$	0	0	0
1	$a_2 + e_5$	e_6	0	0
2	$a_1 + e_4$	e_5	e_6	0
3	$a_0 + e_2$	e_4	e_5	e_6
4	$c_2 + e_2$	$e_6 + e_3$	$e_6 + e_4$	e_5
5	$c_1 + e_1$	$e_5 + e_2$	$e_6 + e_5 + e_3$	$e_6 + e_4$
6	$c_0 + e_0$	$e_6 + e_4 + e_1$	$e_6 + e_5 + e_4 + e_2$	$e_6 + e_5 + e_3$
7	0	$e_6 + e_5 + e_3 + e_0$	$e_5 + e_4 + e_3 + e_1$	$e_6 + e_5 + e_4 + e_2$
		Конфігурація синдрому		
8	0	0	0	0

Визначення тільки бітів помилок для змішування

Рис. 5. Схема декодування



Розглянуті приклади дають змогу в достатній мірі зрозуміти принципи роботи схем декодування лінійного коду.

Далі розглянемо деякі практичні варіанти даного методу декодування.

При декодуванні циклічних кодів з приймального буферного регістра надходить кодова бітова послідовність, що приймається, так, щоб забезпечувалося узгодження операцій виявлення помилкових бітів, синхронізація положень помилкових бітів і виконувалося виправлення помилок. Методи декодування лінійних кодів передбачають запис конфігурацій помилок у комірки пам'яті ПЗП, що відображаються конфігураціями синдрому (адресами ПЗП). Замість цих конфігурацій помилок може бути записана інформація про те, в якому біті за номером відбулася помилка, тобто про місцезнаходження помилкових бітів. Крім того, ці дані про положення помилкових бітів — дані, які зберігаються в

Приймальний буферний регістр								Кер
*RG*0	*RG*1	*RG*2	*RG*3	*RG*4	*RG*5	*RG*6	*RG*7	
0	0	0	0	0	0	0	0	Викл
$a_3 + e_6$	0	0	0	0	0	0	0	
$a_2 + e_5$	$a_3 + e_6$	0	0	0	0	0	0	
$a_1 + e_4$	$a_2 + e_5$	$a_3 + e_6$	0	0	0	0	0	
$a_0 + e_3$	$a_1 + e_4$	$a_2 + e_5$	$a_3 + e_6$	0	0	0	0	
$c_2 + e_2$	$a_0 + e_3$	$a_1 + e_4$	$a_2 + e_5$	$a_3 + e_6$	0	0	0	
$c_1 + e_1$	$c_2 + e_2$	$a_0 + e_3$	$a_1 + e_4$	$a_2 + e_5$	$a_3 + e_6$	0	0	
$c_0 + e_0$	$c_1 + e_1$	$c_2 + e_2$	$a_0 + e_3$	$a_1 + e_4$	$a_2 + e_5$	$a_3 + e_6$	0	Вкл
e_0	e_1	e_2	e_3	e_4	e_5	e_6	Конфігурація помилок	
0	c_0	c_1	c_2	a_0	a_1	a_2	a_3	Викл

У послідовності e_6 — e_0 тільки один біт дорівнює "1"

лінійного коду БЧХ (7,4)

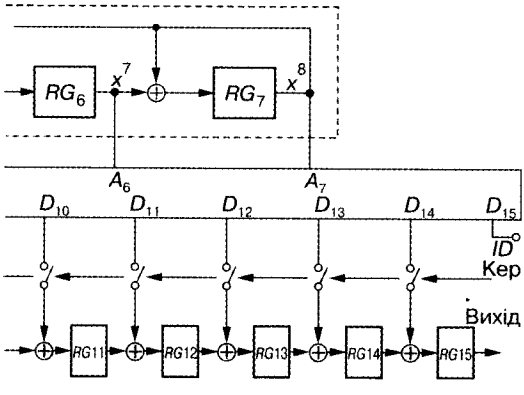


Рис. 6. Схема декодування лінійного коду БЧХ (15, 7)

ПЗП, можна підсумовувати за допомогою лічильника, що дає змогу виконувати виправлення цих помилкових бітів у разі їх надходження на вихід пристрою декодування коду.

16.8. СХЕМИ ДЕКОДУВАННЯ З ВИПРАВЛЕННЯМ t-КРАТНИХ ПОМИЛОК

На рис. 7 наведено схему декодування коду БЧХ (n, k) з виправленням t-кратних помилок, що заснована на принципі ПЗП—лічильник.

Цей принцип полягає в тому, що в ПЗП як дані зберігаються відомості щодо положення помилкових бітів і ці відомості ідентифікуються (підсумовуються) за допомогою лічильників. У схемі (див. рис. 7) у разі обнулення лічильників як корегувальні вихідні сигнали видаються одиниці. Лічильники можуть реалізувати схему як додавання, так і віднімання. Це означає, що на тому такті синхронізації, на якому з'явився сигнал про помилковий біт, вміст лічильника дорівнює нулю, і на його виході формується сигнал, що дорівнює одиниці. Вміст лічильника, тобто число тактів рахунку вибирається так, щоб це число було більше, ніж число кодових бітів на один такт. Крім того, кількість лічильників, потрібних для підрахунку положення помилкового біта, має дорівнювати числу бітів з помилками, що можуть виправлятися.

Для кращого розуміння роботи схеми, поданої на рис. 7, слід враховувати такі чинники:

1. Через M позначено порядок початкового многочлена ($2^M = n + 1$, $n = k + m$).
2. Число розрядів в лічильнику дорівнює 2^M (число розрядів в лічильнику повинно перевищувати число кодових бітів n).
3. У разі використання як лічильника схеми віднімання на вихід ПЗП подаються дані про конкретні номери помилкових бітів 1, ..., n.

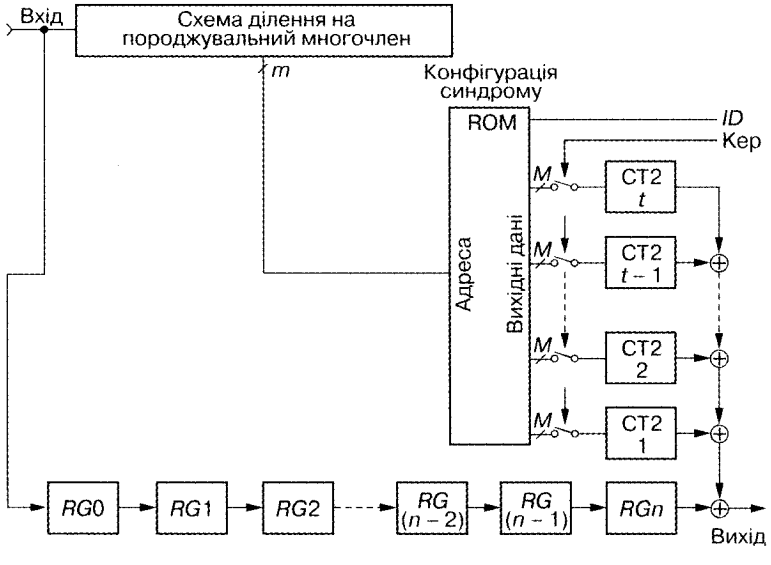


Рис. 7. Схема декодування БЧХ (n, k) коду за принципом ПЗП—лічильник з виправленням t-кратних помилок

4. У разі використання як лічильника схеми додавання на вихід ПЗП подаються дані щодо різниці номерів розрядів лічильника і помилкових бітів 1 ... *n*.

5. За відсутності помилок на вихід ПЗП подається нуль.

6. За наявності нуля на вході лічильника на його виході формується одиниця, а в решті випадків — нуль.

Як приклад розглянемо застосування даного методу виправлення *t*-кратних помилок для декодування коду БЧХ (15, 7) з виправленням, зокрема, подвійних помилок (рис. 8, табл. 2).

Як лічильники використовуються лічильники на базі схеми додавання, а помилки, показані в третьому (a_4) і в п'ятнадцятому бітах (c_0), подаються так:

- помилка в третьому біті — $\overline{a}_4 = a_4 + 1 (\mathrm{mod}\, 2)$,
- помилка в п'ятнадцятому біті — $\overline{c}_0 = c_0 + 1 (\mathrm{mod}\, 2)$.

У табл. 2 процес виправлення помилок показаний з того моменту, коли після повного завершення введення кодової бітової послідовності, що приймається, визначена конфігурація синдрому.

У разі відсутності помилок лічильник встановлюється в "0".

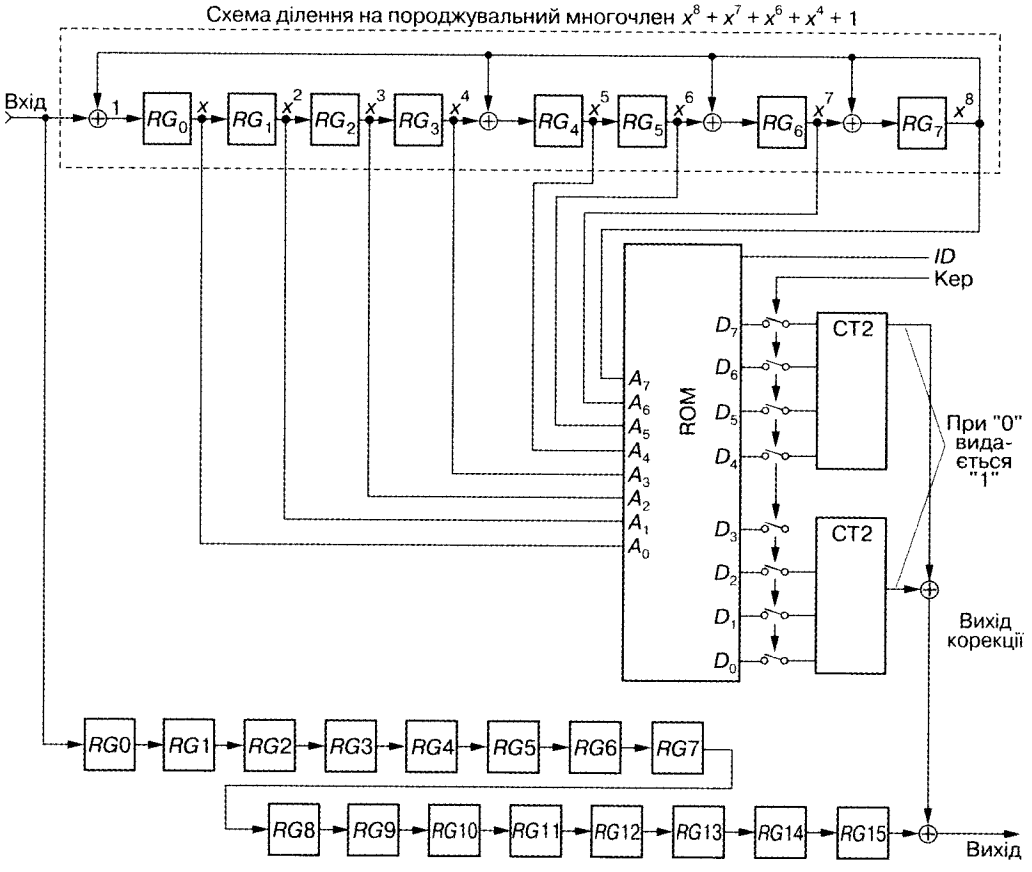


Рис. 8. Схема декодування коду БЧХ (15,7) за принципом ПЗП—лічильник

Таблиця 2. Декодування коду БЧХ (15, 7)

Такт	Вхід	Виходи приймального буферного регістра						Конфігурація синдрому							
		$RG0$	$RG1$	...	$RG13$	$RG14$	$RG15$	RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7
0	0	$\bar{c}_0$	c_1	...	a_5	a_6	0	1	1	0	1	1	1	0	0
1	0	0	$\bar{c}_0$	...	$\bar{a}_4$	a_5	a_6								
2	0	0	0	...	a_3	$\bar{a}_4$	a_5								
3	0	0	0	...	a_2	a_3	$\bar{a}_4$								
4	0	0	0	...	a_1	a_2	a_3								
5	0	0	0	...	a_0	a_1	a_2								
6	0	0	0	...	c_7	a_0	a_1								
7	0	0	0	...	c_6	c_7	a_0								
8	0	0	0	...	c_5	c_6	c_7								
9	0	0	0	...	c_4	c_5	c_6								
10	0	0	0	...	c_3	c_4	c_5								
11	0	0	0	...	c_2	c_3	c_4								
12	0	0	0	...	c_1	c_2	c_3								
13	0	0	0	...	$\bar{c}_0$	c_1	c_2								
14	0	0	0	...	0	$\bar{c}_0$	c_1								
15	0	0	0	...	0	0	$\bar{c}_0$								

Конфігурація синдрому при помилках у 3-му і 15-му бітах

D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	1	1	0	1	0	0	0	1

$ID = 0$

Встановлення лічильника 2

2^3	2^2	2^1	2^0
1	1	0	1

Помилка у 3-му біті

Встановлення лічильника 1

2^3	2^2	2^1	2^0
0	0	0	1

Помилка у15-му біті

Таблиця 3. Декодування коду БЧХ

Такт	Вхід	Виходи приймального буферного регістра						Конфігурація синдрому							
		$RG0$	$RG1$	...	$RG13$	$RG14$	$RG15$	RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7
0	0	c_0	c_1	...	a_5	a_6	0	0	0	0	0	0	0	0	0
1	0	0	c_0	...	a_4	a_5	a_6								
2	0	0	0	...	a_3	a_4	a_5								
3	0	0	0	...	a_2	a_3	a_4								
4	0	0	0	...	a_1	a_2	a_3								
5	0	0	0	...	a_0	a_1	a_2								
6	0	0	0	...	c_7	a_0	a_1								
7	0	0	0	...	c_6	c_7	a_0								
8	0	0	0	...	c_5	c_6	c_7								
9	0	0	0	...	c_4	c_5	c_6								
10	0	0	0	...	c_3	c_4	c_5								
11	0	0	0	...	c_2	c_3	c_4								
12	0	0	0	...	c_1	c_2	c_3								
13	0	0	0	...	c_0	c_1	c_2								
14	0	0	0	...	0	c_0	c_1								
15	0	0	0	...	0	0	c_0								

Конфігурація синдрому за відсутності помилок (адреса ПЗП)

D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0
0	0	0	0	0	0	0	0	0

$ID = 0$

Встановлення лічильника 2

2^3	2^2	2^1	2^0
0	0	0	0

Відсутність помилок

Встановлення лічильника 1

2^3	2^2	2^1	2^0
0	0	0	0

Відсутність помилок

з виправленням подвійних помилок

Лічильник 1				Лічильник 2				Вихід корекції	Вихід (вихід корекції + вихід *RG*15)
2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0		
Встановлення лічильників								—	—
0	0	0	1	1	1	0	1		
0	0	1	0	1	1	1	0	0	$a_6 + 0 \rightarrow a_6$
0	0	1	1	1	1	1	1	0	$a_5 + 0 \rightarrow a_5$
0	1	0	0	0	0	0	0	1	$\bar{a}_4 + 1 \rightarrow a_4$
0	1	0	1	0	0	0	1	0	$a_3 + 0 \rightarrow a_3$
0	1	1	0	0	0	1	0	0	$a_2 + 0 \rightarrow a_2$
0	1	1	1	0	0	1	1	0	$a_1 + 0 \rightarrow a_1$
1	0	0	0	0	1	0	0	0	$a_0 + 0 \rightarrow a_0$
1	0	0	1	0	1	0	1	0	$c_7 + 0 \rightarrow c_7$
1	0	1	0	0	1	1	0	0	$c_6 + 0 \rightarrow c_6$
1	0	1	1	0	1	1	1	0	$c_5 + 0 \rightarrow c_5$
1	1	0	0	1	0	0	0	0	$c_4 + 0 \rightarrow c_4$
1	1	0	1	1	0	0	1	0	$c_3 + 0 \rightarrow c_3$
1	1	1	0	1	0	1	0	0	$c_2 + 0 \rightarrow c_2$
1	1	1	1	1	0	1	1	0	$c_1 + 0 \rightarrow c_1$
0	0	0	0	1	1	0	0	1	$\bar{c}_0 + 1 \rightarrow c_0$

(15,7) за відсутності помилок

Лічильник 1				Лічильник 2				Вихід корекції	Вихід (вихід корекції + + вихід *RG*15)
2^3	2^2	2^1	2^0	2^3	2^2	2^1	2^0		
Встановлення лічильників								—	—
0	0	0	0	0	0	0	0		
0	0	0	1	0	0	0	1	0	$a_6 + 0 \rightarrow a_6$
0	0	1	0	0	0	1	0	0	$a_5 + 0 \rightarrow a_5$
0	0	1	1	0	0	1	1	0	$a_4 + 0 \rightarrow a_4$
0	1	0	0	0	1	0	0	0	$a_3 + 0 \rightarrow a_3$
0	1	0	1	0	1	0	1	0	$a_2 + 0 \rightarrow a_2$
0	1	1	0	0	1	1	0	0	$a_1 + 0 \rightarrow a_1$
0	1	1	1	0	1	1	1	0	$a_0 + 0 \rightarrow a_0$
1	0	0	0	1	0	0	0	0	$c_7 + 0 \rightarrow c_7$
1	0	0	1	1	0	0	1	0	$c_6 + 0 \rightarrow c_6$
1	0	1	0	1	0	1	0	0	$c_5 + 0 \rightarrow c_5$
1	0	1	1	1	0	1	1	0	$c_4 + 0 \rightarrow c_4$
1	1	0	0	1	1	0	0	0	$c_3 + 0 \rightarrow c_3$
1	1	0	1	1	1	0	1	0	$c_2 + 0 \rightarrow c_2$
1	1	1	0	1	1	1	0	0	$c_1 + 0 \rightarrow c_1$
1	1	1	1	1	1	1	1	0	$c_0 + 0 \rightarrow c_0$

16.9. ЛОГІЧНІ СХЕМИ ОБЧИСЛЕННЯ СИНДРОМУ

Під час надходження на вихід кодової бітової послідовності, що приймається, число імпульсів підрахунку вибирається більшим, ніж число кодових бітів. Отже, таких ситуацій, за яких вміст лічильника дорівнюватиме нулю, не виникає (табл. 3).

У наведеній вище схемі синдром визначався з використанням схеми ділення на породжувальний многочлен. Крім того, цей синдром можна отримати за логічною схемою за допомогою обчислення матриці перевірки на парність і кодової бітової послідовності, що приймається.

Матриця контролю парності коду БЧХ (7, 4) має такий вигляд:

$$H = \begin{pmatrix} 1110100 \\ 0111010 \\ 1101001 \end{pmatrix}.$$

Якщо рядок кодової бітової послідовності, що приймається, записується у вигляді

$$u = (a_3', a_2', a_1', a_0', c_2', c_1', c_0'),$$

то синдром помилки можна подати так:

$$S = \begin{pmatrix} S_2 \\ S_1 \\ S_0 \end{pmatrix} = Hu^T = \begin{pmatrix} 1110100 \\ 0111010 \\ 1101001 \end{pmatrix} \begin{pmatrix} a_3' \\ a_2' \\ a_1' \\ a_0' \\ c_2' \\ c_1' \\ c_0' \end{pmatrix} = \begin{pmatrix} a_3' + a_2' + a_1' + c_2' \\ a_2' + a_1' + a_0' + c_1' \\ a_3' + a_2' + a_0' + c_0' \end{pmatrix}. \quad (1)$$

Згідно з виразом (1), кожний біт синдрому S можна записати як систему рівнянь:

$$\left.\begin{aligned} S &= a_3' + a_2' + a_1' + c_2' \ (\text{mod } 2), \\ S &= a_2' + a_1' + a_0' + c_1' \ (\text{mod } 2), \\ S &= a_3' + a_2' + a_0' + c_0' \ (\text{mod } 2). \end{aligned}\right\} \quad (2)$$

Відповідно до цієї системи, схема для обчислення синдрому S має вигляд, наведений на рис. 9. Як бачимо, синдром обчислюється до того моменту, коли

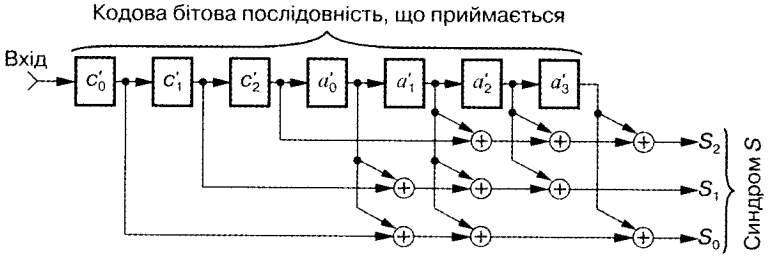


Рис. 9. Логічна схема обчислення синдрому коду БЧХ (7, 4)

закінчується введення кодової бітової послідовності, що приймається, тобто він з'являється на приймальному боці обчислювальної схеми.

У разі коду БЧХ (15, 7) матриця контролю парності за стовпцями має вигляд

$$H = \begin{pmatrix} 100010110000000 \\ 110011101000000 \\ 111011000100000 \\ 011101100010000 \\ 101100000001000 \\ 010110000000100 \\ 001011000000010 \\ 000101100000001 \end{pmatrix}.$$

Отже, якщо загальну кодову бітову послідовність із інформаційних і контрольних бітів, що приймаються, подати у вигляді рядка

$$u = (a_6', a_5', a_4', a_3', a_2', a_1', a_0', c_7', c_6', c_5', c_4', c_3', c_2', c_1', c_0'),$$

то синдром S при цьому можна записати так:

$$S = \begin{pmatrix} S_7 \\ S_6 \\ S_5 \\ S_4 \\ S_3 \\ S_2 \\ S_1 \\ S_0 \end{pmatrix} = Hu^T.$$

Тому, якщо виконати обчислення за формулою (1), то відповідні біти синдрому S можна записати так:

$$\left.\begin{aligned} S_7 &= a_6' + a_2' + a_0' + c_7' && (\mathrm{mod}\,2), \\ S_6 &= a_6' + a_5' + a_2' + a_1' + a_0' + c_6' && (\mathrm{mod}\,2), \\ S_5 &= a_6' + a_5' + a_4' + a_2' + a_1' + c_5' && (\mathrm{mod}\,2), \\ S_4 &= a_5' + a_4' + a_3' + a_1' + a_0' + c_4' && (\mathrm{mod}\,2), \\ S_3 &= a_6' + a_4' + a_3' + c_3' && (\mathrm{mod}\,2), \\ S_2 &= a_5' + a_3' + a_2' + c_2' && (\mathrm{mod}\,2), \\ S_1 &= a_4' + a_2' + a_1' + c_1' && (\mathrm{mod}\,2), \\ S_0 &= a_3' + a_1' + a_0' + c_0' && (\mathrm{mod}\,2). \end{aligned}\right\} \quad (3)$$

Принципова схема для обчислення синдрому зображена на рис. 10.

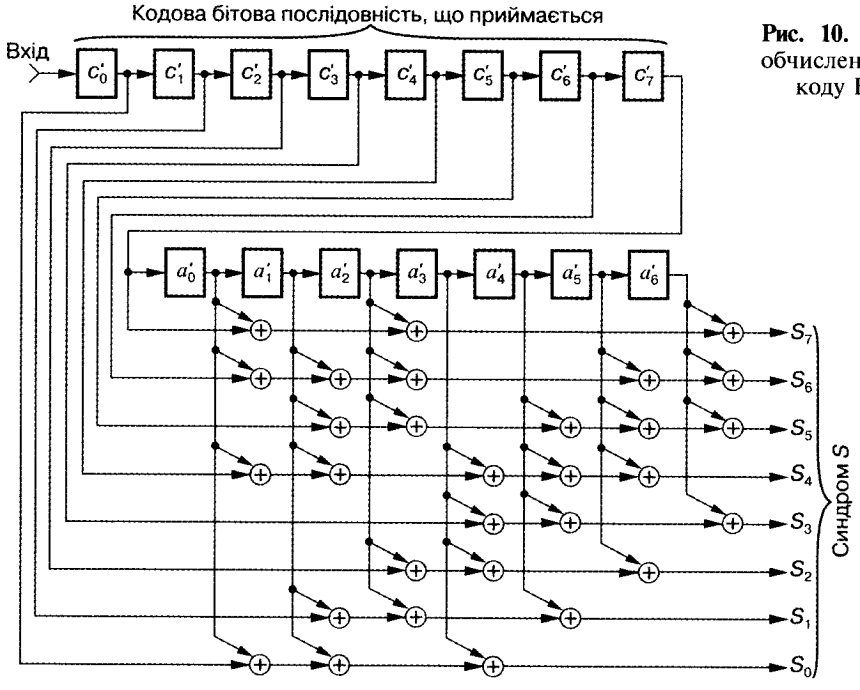


Рис. 10. Логічна схема обчислення синдрому коду БЧХ (15, 7)

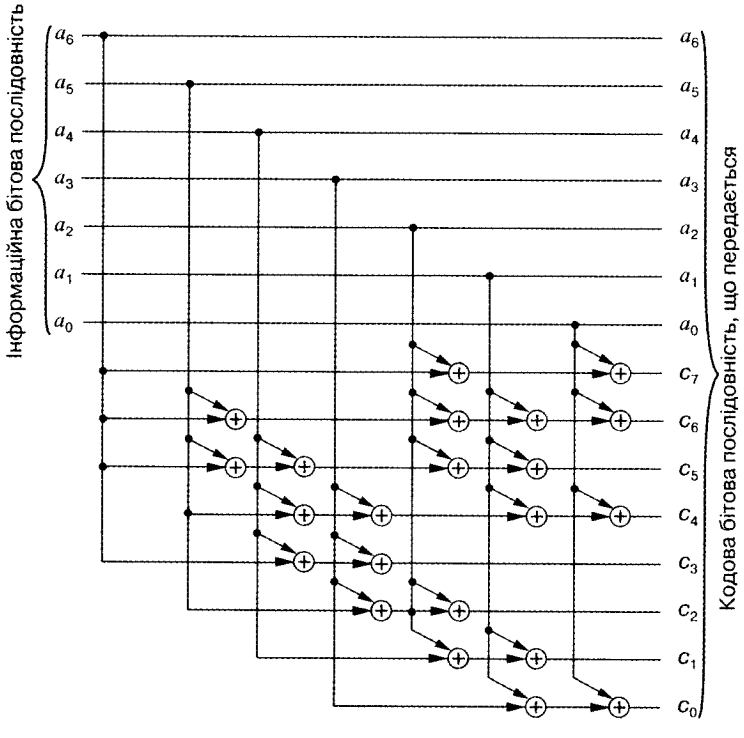


Рис. 11. Схема кодування коду БЧХ (15, 7) як лінійного коду

Наведена схема для обчислення синдрому коду БЧХ (15, 7) реалізована відповідно до формули (3).

Робота схеми обчислення синдрому здійснюється наступним чином. Кодова комбінація, що приймається, подається одночасно в блок регістрів і в обчислювач синдрому, при цьому відлік йде від старшого розряду. Після завершення перших n тактів блок регістрів буде повністю заповнений, а в обчислювачі синдрому буде виконано ділення прийнятої комбінації на породжувальний многочлен. Якщо залишок від ділення дорівнює нулю, то на виходах обчислювача синдрому будуть нульові потенціали. У цьому разі на $(n + 1)$-му такті обчислювач синдрому видає інформацію на вихід пристрою. Якщо хоча б одна компонента коду синдрому відмінна від нуля, то інформація не видається. Оскільки етап виправлення помилок кодової комбінації, що приймається, збігається з етапом обчислення синдрому подальшої кодової комбінації, то імпульси зчитування інформації повторюються через кожні n тактів.

Мабуть, у разі кодування циклічного коду як лінійного застосовувати лише операцію обчислення синдрому за допомогою схеми ділення на породжувальний многочлен недоцільно, навіть якщо замість схем ділення на породжувальний многочлен, поданих на рис. 5 та 6, використовувати досконаліші схеми (див. рис. 9 та 10). Це пояснюється тим, що в цьому випадку при кодуванні циклічного коду абсолютно не використовуються всі його позитивні властивості.

У разі кодування циклічного коду як лінійного коду використовувалися інформаційна бітова послідовність і породжувальна матриця. Застосовуючи описану вище процедуру і схеми додавання за модулем 2, нескладно реалізувати логічну схему кодування на базі цієї інформаційної бітової послідовності і породжувальної матриці. Наприклад, схему кодування коду БЧХ (15, 7) як лінійного коду можна подати у вигляді, наведеному на рис. 11.

Розділ 17

ДЕКОДУВАННЯ ЦИКЛІЧНИХ КОДІВ ЗА ДИСТАНЦІЙНИМ ПРИНЦИПОМ

Відомо, що оскільки циклічні коди можна подати у вигляді лінійних кодів, то і декодувати їх можна за принципами декодування лінійних кодів. Тому враховуючи властивості циклічних кодів, помилкові біти можна виправляти, використовуючи синдром (ознаку), що отримуємо як залишок породжувального многочлена. При цьому використовуються табличний пошук і ПЗП, у якому з'ясовано конфігурацію помилки, що відповідає цьому синдрому.

На підставі властивостей схем ділення на многочлен, що породжує і відображає синдром залишку, доцільно дати оцінку й іншому принципу декодування циклічних кодів, який називається *дистанційним.*

17.1. ФУНКЦІЇ СХЕМ ДІЛЕННЯ НА ПОРОДЖУВАЛЬНИЙ МНОГОЧЛЕН

Розглянемо детальніше власне схеми ділення на многочлен, що породжує циклічний код, використовуючи їх зв'язок зі способом декодування лінійних кодів.

Як приклад знову розглянемо код БЧХ (7, 4). У цьому коді, призначеному для лінійного кодування, використовуються інформаційна послідовність

$$a = (a_3,\ a_2,\ a_1,\ a_0)$$

і породжувальна матриця

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix}.$$

Кодова послідовність, що передається, має вигляд

$$v = aG = (a_3, a_2, a_1, a_0, c_2, c_1, c_0),$$

де

$$c_2 = a_3 + a_2 + a_1 \pmod 2;$$

$$c_1 = a_2 + a_1 + a_0 \pmod 2;$$

$$c_0 = a_3 + a_2 + a_0 \pmod 2.$$

Оскільки лінійний код — це група (послідовність), то даний код БЧХ (7, 4) є групою з числом кодових векторів $2^7 = 128$ (число основ груп), кодових слів $2^4 = 16$ (число основ підгруп) і досліджуваних станів $2^3 = 8$ (число класів залишків). Спробуємо розвернути ці кодові вектори в класи залишків за допомогою кодових слів, що є підгрупами, і створити стандартний масив коду БЧХ (7, 4).

Цей код належить до кодів з виправленням одиничних помилок, тому конфігурації помилок аж до одиничних помилок, тобто конфігурації без помилок і конфігурації з одиничними помилками, можуть використовуватися як типові основи при формуванні стандартних масивів. Спробуємо визначити число конфігурацій помилок аж до конфігурації одиничної помилки, що виправляється.

Зазвичай, в послідовності, що складається з кодових слів завдовжки n бітів, якщо виникає t-бітна помилка, то число кодових слів, в яких з'являються t помилок (помилок вагою t), дорівнює числу правильних кодових слів та числу з'єднань по t і може визначатися за формулою

$$\left(\frac{n}{t}\right) = \frac{n!}{t!(n-t)!}.$$

Іншими словами, оскільки число кодових слів, в яких до кодових слів лінійного (n, k) коду $a_{k-1}, a_{k-2}, \ldots, a_0, c_{m-1}, c_{m-2}, \ldots, c_0$ додано конфігурацію помилок вагою t, дорівнює числу $\left(\frac{n}{t}\right)$ з'єднань по t, то число конфігурацій помилок вагою t становить

$$\left(\frac{n}{t}\right) = \frac{n!}{t!(n-t)!}. \qquad (4)$$

Факторіал n обчислюється так:

$$n! = n \cdot (n-1) \cdot (n-2) \cdot \ldots \cdot 2 \cdot 1,$$

де $0! = 1$. Виконавши цю підстановку, одержимо

$$\left(\frac{n}{t}\right) = \frac{n \cdot (n-1) \cdot \ldots \cdot (n-t+1)}{t \cdot (t-1) \cdot \ldots \cdot 2 \cdot 1 \cdot (n-t) \cdot (n-t-1)} = \frac{n \cdot (n-1) \cdot \ldots \cdot (n-t+1)}{t \cdot (t-1) \cdot \ldots \cdot 2 \cdot 1}. \qquad (5)$$

Довжина кодових слів коду БЧХ (7, 4) — $n = 7$, навіть і у разі відсутності помилок в кодовому слові: $t = 0$. Отже, число безпомилкових конфігурацій слова дорівнює

$$\left(\frac{7}{0}\right) = \frac{7!}{0!7!} = 1.$$

Кодове слово, що містить одиничну помилку, відрізняється від правильного кодового слова тільки на 1 біт, тому якщо $t = 1$, то це число дорівнює

$$\left(\frac{7}{1}\right) = \frac{7!}{1!6!} = 7.$$

Отже, число конфігурацій слів з одиничною помилкою дорівнюватиме 7. Зрозуміло, що число конфігурацій помилок, які виправляються, становить

(*число безпомилкових конфігурацій*) + (*число конфігурацій одиничних помилок*) = $\left(\frac{7}{0}\right)+\left(\frac{7}{1}\right)=1+7=8$.

Оскільки в коді БЧХ (7, 4) є вісім досліджуваних станів, тобто вісім класів залишків, то число конфігурацій помилок, що виправляються, менше ніж 8 і всі конфігурації помилок, що виправляються, можна вважати типовими основами класів вирахувань. Число конфігурацій помилок дорівнює 8 і записується так:

(0 0 0 0 0 0 0) — безпомилкова конфігурація

(1 0 0 0 0 0 0)
(0 1 0 0 0 0 0)
(0 0 1 0 0 0 0)
(0 0 0 1 0 0 0) — конфігурації з одиничною помилкою.
(0 0 0 0 1 0 0)
(0 0 0 0 0 1 0)
(0 0 0 0 0 0 1)

Стандартні масиви коду БЧХ (7, 4), що є типовими основами, наведені в табл. 4.

Таблиця 4. Стандартні масиви коду БЧХ (7, 4)

Інформаційна кодова послідовність	0000	0001	0010	0011	0100	0101	0110	0111
Кодова послідовність, що передається	0000000	0001011	0010110	0011101	0100111	0101100	0110001	0111010
Конфігурація помилок	Кодова послідовність,							
0000000	0000000	0001011	0010110	0011101	0100111	0101100	0110001	0111010
1000000	1000000	1001011	1010110	1011101	1100111	1101100	1110001	1111010
0100000	0100000	0101011	0110110	0111101	0000111	0001100	0010001	0011010
0010000	0010000	0011011	0000110	0001101	0110111	0111100	0100001	0101010
0001000	0001000	0000011	0011110	0010101	0101111	0100100	0111001	0110010
0000100	0000100	0001111	0010010	0011001	0100011	0101000	0110101	0111110
0000010	0000010	0001001	0010100	0011111	0100101	0101110	0110011	0111000
0000001	0000001	0001010	0010111	0011100	0100110	0101101	0110000	0111011

При передачі інформації конфігурації помилок додаються до інформаційних кодових слів. Конфігурація помилки, що виправляється, усередині конфігурації помилки, доданої до кодового слова, як з'ясувалося, є лише конфігурацією помилки, відібраної як типова основа. У разі, коли до кодового слова додається конфігурація крім помилкової, яка була відібрана як типова основа, виправлення помилки, тобто декодування, неможливе, оскільки перевищено можливість виправлення помилки, що міститься в коді.

У кодовій послідовності, що приймається, як наведено вище, конфігурація помилки $e = (e_6, e_5, e_4, e_3, e_2, e_1, e_0)$ (в інтервалі e_6—e_0 тільки один член дорівнює 1) додається до кодового слова v і формується кодова послідовність, що приймається, у такому вигляді:

$$u = v + e = (a_3 + e_6, a_2 + e_5, a_1 + e_4, a_0 + e_3, c_2 + e_2, c_1 + e_1, c_0 + e_0) \text{ (mod 2)}.$$

Таким чином, за цією кодовою послідовністю u, що приймається, і матрицею контролю парності $H = (\alpha^6, \alpha^5, \alpha^4, \alpha^3, \alpha^2, \alpha^1, \alpha^0)$ обчислюється синдром $S = Hu^T = He^T$, а також виконується виправлення помилок унаслідок додавання конфігурації помилок, яка відповідає цьому синдрому, до кодової послідовності, що приймається. У табл. 5 наведено синдроми і відповідні їм конфігурації помилок для цього прикладу.

Спробуємо тепер декодувати цей код БЧХ (7, 4), вважаючи його циклічним кодом.

Код БЧХ (7, 4) як циклічний код, як і код Хеммінга (7, 4) (див. ч. 1), є кодом, отриманим з використанням породжувального многочлена $G(x) = x^3 + x + 1$.

На приймальному кінці у разі циклічних кодів можна автоматично обчислити синдром, використовуючи схему ділення на породжувальний многочлен, як залишок породжувального многочлена. Крім того, використовуючи схеми

1000	1001	1010	1011	1100	1101	1110	1111
1000101	1001110	1010011	1011000	1100010	1101001	1110100	1111111
що приймається							
1000101	1001110	1010011	1011000	1100010	1101001	1110100	1111111
0000101	0001110	0010011	0011000	0100010	0101001	0110100	0111111
1100101	1101110	1110011	1111000	1000010	1001001	1010100	1011111
1010101	1011110	1000011	1001000	1110010	1111001	1100100	1101111
1001101	1000110	1011011	1010000	1101010	1100001	1111100	1110111
1000001	1001010	1010111	1011100	1100110	1101101	1110000	1111011
1000111	1001100	1010001	1011010	1100000	1101011	1110110	1111101
1000100	1001111	1010010	1011001	1100011	1101001	1110101	1111110

Таблиця 5. Синдроми коду БЧХ (7, 4) і відповідні їм конфігурації помилок

Синдром S	Конфігурація помилок
0	0000000
α^6	1000000
α^5	0100000
α^4	0010000
α^3	0001000
α^2	0000100
α^1	0000010
α^0	0000001

ділення, як вони є, і завдяки багатократному діленню можна визначити конфігурацію заданого залишку і виправити помилковий біт (табл. 6).

Розглянемо детально ці операції. Якщо необхідно визначити залишок конфігурації помилки e за допомогою схеми ділення на породжувальний многочлен $x^3 + x + 1$, то у момент закінчення введення сьомого біта це вирахування $R(\alpha)$ визначається так:

$$R(\alpha) = (e_6 + e_5 + e_4 + e_2)\alpha^2 + (e_5 + e_4 + e_3 + e_1)\alpha^1 + (e_6 + e_5 + e_3 + e_0)\alpha^0 \pmod 2. \quad (6)$$

Тут α — корінь початкового многочлена $x^3 + x + 1$, тобто вихідна основа поля $GF(2^3)$. Як видно з табл. 6, з рівняння (6) випливають конфігурації помилок і залишків, а отже, і співвідношення синдромів.

На рис. 12 наведено схему декодування коду БЧХ (7, 4) з моменту закінчення введення сьомого біта.

У момент закінчення введення сьомого біта, тобто після завершення повного введення матриці коду, що приймається, момент визначення залишку як синдрому помилки розглядається як нульовий такт.

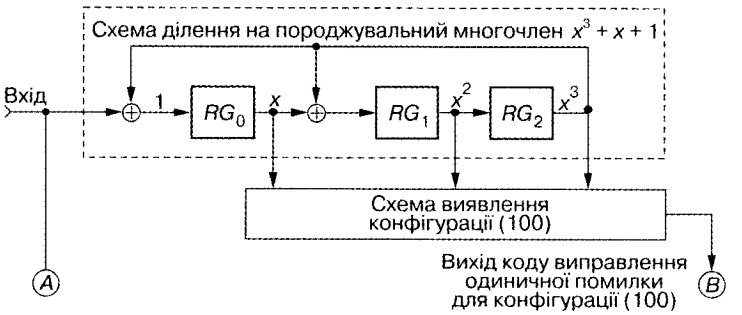
Отже, у момент нульового такту на виходах регістра $RG0$, $RG1$, $RG2$ схеми ділення на породжувальний многочлен формується залишок, що ділиться на породжувальний многочлен $x^3 + x + 1$ конфігурації помилки e. Після цього на вхід подається "0". Якщо використовувати схему декодування цього стану і властивість періодичності залишку, то можна виправити помилку. Оскільки виведення сигналів виправлення помилок для формування конфігурації синдрому (100) виконується по тактах (по черзі) у такій послідовності:

1 такт → виведення сигналу виправлення помилки першого біта,
2 такт → виведення сигналу виправлення помилки другого біта,
3 такт → виведення сигналу виправлення помилки третього біта,
4 такт → виведення сигналу виправлення помилки четвертого біта,
5 такт → виведення сигналу виправлення помилки п'ятого біта,
6 такт → виведення сигналу виправлення помилки шостого біта,
7 такт → виведення сигналу виправлення помилки сьомого біта,

Таблиця 6. Конфігурації помилок і залишок

Конфігурація помилок	Залишок	Основа поля $GF(2^3)$
(0000000) ($e_6 - e_0 = 0$)	0	0
(1000000) ($e_6 = 1$)	$\alpha^2 +$ + 1	α^6
(0100000) ($e_5 = 1$)	$\alpha^2 + \alpha^1 + 1$	α^5
(0010000) ($e_4 = 1$)	$\alpha^2 + \alpha^1$	α^4
(0001000) ($e_3 = 1$)	$\alpha^1 + 1$	α^3
(0000100) ($e_2 = 1$)	α^2	α^2
(0000010) ($e_1 = 1$)	α^1	α^1
(0000001) ($e_0 = 1$)	1	α^0

Декодування циклічних кодів за дистанційним принципом



Такт	Вихід RG_0	Вихід RG_1	Вихід RG_2	Вихід конфігурації (100)	Вихід конфігурації (100)
0	$e_6 + e_5 + e_3 + \langle e_0 \rangle$	$e_5 + e_4 + e_3 + e_1$	$e_6 + e_5 + e_4 + e_2$	e_0	(0000001)
1	$\langle e_6 \rangle + e_5 + e_4 + e_2$	$e_4 + e_3 + e_2 + e_0$	$e_5 + e_4 + e_3 + e_1$	e_6	(1000000)
2	$\langle e_5 \rangle + e_4 + e_3 + e_1$	$e_6 + e_3 + e_2 + e_1$	$e_4 + e_3 + e_2 + e_0$	e_5	(0100000)
3	$\langle e_4 \rangle + e_3 + e_2 + e_0$	$e_5 + e_2 + e_1 + e_0$	$e_6 + e_3 + e_2 + e_1$	e_4	(0010000)
4	$e_6 + \langle e_3 \rangle + e_2 + e_1$	$e_6 + e_4 + e_1 + e_0$	$e_5 + e_2 + e_1 + e_0$	e_3	(0001000)
5	$e_5 + \langle e_2 \rangle + e_1 + e_0$	$e_6 + e_5 + e_3 + e_0$	$e_6 + e_4 + e_1 + e_0$	e_2	(0000100)
6	$e_6 + e_4 + \langle e_1 \rangle + e_0$	$e_6 + e_5 + e_4 + e_2$	$e_6 + e_5 + e_3 + e_0$	e_1	(0000010)
7	$e_6 + e_5 + e_3 + \langle e_0 \rangle$	$e_5 + e_4 + e_3 + e_1$	$e_6 + e_5 + e_4 + e_2$	e_0	(0000001)

У послідовності e_6–e_0 тільки один біт дорівнює "1"

Зміна залишку до виправлення e_0	Конфігурація помилки, що відповідає залишку	Основа поля $GF(2^3)$
1	(0000001)	1
α	(0000010)	α
α^2	(0000100)	α^2
$\alpha + 1$	(0001000)	α^3
$\alpha^2 + \alpha$	(0010000)	α^4
$\alpha^2 + \alpha + 1$	(0100000)	α^5
$\alpha^2 + 1$	(1000000)	α^6
1	(0000001)	α^7 (=1)

$e_0 = 1$, інші "0"

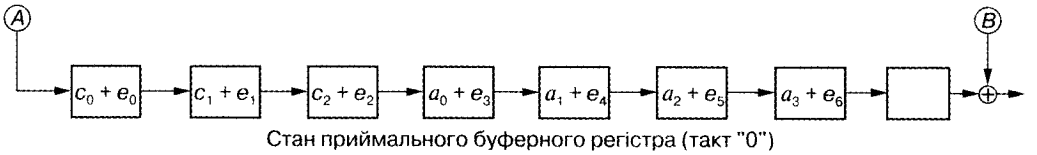


Рис. 12. Схема декодування коду БЧХ (7, 4)

то і в приймальному буферному регістрі накопичується ряд кодів, що приймаються, в такій самій послідовності. Тобто в приймальному буферному регістрі коди, що поступають, автоматично розташовуються в ряд від першого до сьомого біта.

Після першого такту, оскільки здійснюється виведення сигналу виправлення помилки першого біта, з моменту закінчення повного введення коду, що приймається, для зіставлення часових інтервалів приймального буферного регістра розглянемо ще один розряд буферного регістра. У разі декодування подібно до того, як після першого такту виводиться сигнал виправлення помилки першого біта, а після другого такту виводиться сигнал виправлення помилки другого біта, виводиться конфігурація помилок і виправляється помилковий біт на виході приймального буферного регістра (табл. 7).

Т а б л и ц я 7. Сигнали виправлення помилок при декодуванні

Параметр	Такт							
	0	1	2	3	4	5	6	7
Вихід виправлення помилки	e_0	e_6	e_5	e_4	e_3	e_2	e_1	e_0
Додавання за модулем два	—	+	+	+	+	+	+	+
Вихід приймального буферного регістра	—	$a_3 + e_6$	$a_2 + e_5$	$a_1 + e_4$	$a_0 + e_3$	$c_2 + e_2$	$c_1 + e_1$	$c_0 + e_0$

Зауважимо, що при такті "0" із схеми виявлення конфігурації помилок здійснюється виведення сигналу виправлення помилки в сьомому біті e_0, тобто до моменту закінчення введення послідовності кодів, що приймаються.

Такий докладний аналіз сигналів виправлення помилок у разі декодування виконаний з єдиною метою — вибрати спосіб декодування за дистанційним принципом.

У момент такту "0", якщо перший біт помилковий ($e_6 = 1$, інші — "0"), залишок дорівнюватиме $\alpha^2 + 1$ (див. табл. 6). Після першого такту ця помилка в першому біті (e_6) виявляється, і за допомогою сигналу виправлення помилки вона виправляється. Аналогічно, у разі помилки в другому біті ($e_5 = 1$, інші — "0") ця помилка виявляється, а після другого такту виводиться сигнал виправлення помилки і виправляється помилка у другому біті e_5:

Такт	Залишок
0	+1
1	1

← Виявлена конфігурація (100)

Такт	Залишок
0	$\alpha^2 + \alpha + 1$
1	$\alpha^2 + 1$
2	1

← Виявлена конфігурація (100)

Для того щоб виявити стан, за якого дорівнюють одиниці відповідні помилкові біти (виявлена конфігурація (100)) у разі змін за кожним тактом залишків, розглянемо зміну залишків у разі помилки в сьомому біті e_0, який виправляється останнім.

У правій частині таблиці (див. рис. 12) наведено операції визначення залишків у сьомому біті e_0 і конфігурації помилок, які відповідають цим залишкам. Зміни основи поля $GF(2^3)$, що є одним з виразів залишку, поданого в цій таблиці, мають вигляд:

Такт	0	1	2	3	4	5	6	7
Основа поля $GF(2^3)$	$\alpha^0 = 1$	α	α^2	α^3	α^4	α^5	α^6	$\alpha^7 = 1$

Звідси стають зрозумілими зміни, яких зазнає вихідна основа поля $GF(2^3)$ по кожному такту, якщо спрацьовує схема ділення на породжувальний многочлен, коли основа поля $GF(2^3)$ дорівнює одиниці (при "0" на вході схеми

ділення). Отже, для конкретного залишку схема ділення на породжувальний многочлен функціонуватиме як схема множення на початкову основу α.

Далі виразимо залишок породжувального многочлена за допомогою кінцевої основи. У табл. 6 наведено співвідношення залишку породжувального многочлена $x^3 + x + 1$ та основи поля $GF(2^3)$, і якщо залишок $R(\alpha)$ при введенні конфігурації помилки подати, використовуючи це співвідношення, то

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e\alpha + e_0. \quad (7)$$

Оскільки у рівнянні (7) для $R(\alpha)$ α є коренем виразу $x^3 + x + 1$, то $R(\alpha)$ фактично, як і в рівнянні (6), є многочленом з α у степені меншому ніж 2. Проте спосіб формульного подання за допомогою основи поля $GF(2^3)$ як залишку $R(\alpha)$у рівнянні (7) є надзвичайно ефективним.

У кінцевий момент введення многочлена коду, що приймається, визначається як залишок $R(\alpha)$, поданий у рівнянні (7) як синдром. Визначивши цей залишок $R(\alpha)$, коли реалізовано схему ділення на породжувальний многочлен, яка функціонує як схема множення, з'ясуємо ознаки, яких набуває α у залишку $R(\alpha)$ у рівнянні (7). Проте, з огляду на те, що α є початковою (вихідною) основою поля $GF(2^3)$, справедливою є рівність

$$\alpha^7 = 1.$$

Подібно до того, як виконується перехід

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0,$$
$$\alpha R(\alpha) = e_5\alpha^6 + e_4\alpha^5 + e_3\alpha^4 + e_2\alpha^3 + e_1\alpha^2 + e_0\alpha + e_6,$$
$$\alpha^2 R(\alpha) = e_4\alpha^6 + e_3\alpha^5 + e_2\alpha^4 + e_1\alpha^3 + e_0\alpha^2 + e_6\alpha + e_5,$$
$$\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots$$
$$\alpha^6 R(\alpha) = e_0\alpha^6 + e_6\alpha^5 + e_5\alpha^4 + e_4\alpha^3 + e_3\alpha^2 + e_2\alpha + e_1,$$
$$\alpha^7 R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0 = R(\alpha),$$

α^7 послідовно переходить в α. Отже, після закінчення семи тактів конфігурація синдрому схеми ділення набуває первинного вигляду. У цьому випадку під час виконання одного такту схеми ділення конфігурація помилки e, виражена за допомогою залишку, змінюється, як показано нижче:

такт 0 — $(e_6, e_5, e_4, e_3, e_2, e_1, e_0)$,
такт 1 — $(e_5, e_4, e_3, e_2, e_1, e_0, e_6)$,
такт 2 — $(e_4, e_3, e_2, e_1, e_0, e_6, e_5)$,
.....
такт 6 — $(e_0, e_6, e_5, e_4, e_3, e_2, e_1)$,
такт 7 — $(e_6, e_5, e_4, e_3, e_2, e_1, e_0)$.

За такої схеми ділення конфігурація помилки здійснює повний обхід і після семи тактів завершує один цикл (рис. 13).

У правій частині таблиці (див. рис. 12) наведено конфігурації одиничної помилки в сьомому біті (0000001) ($e_0 = 1$, інші — "0"), що виявляє ознаки, наявні у разі циклічних потактових переходів. Як бачимо, якщо реалізується залишок одиничної помилки (0000001), визначений у регістрі зсуву схеми ділення, то повний цикл здійснюється за сім тактів, і в межах одного циклу виявляється повна конфігурація одиничної помилки.

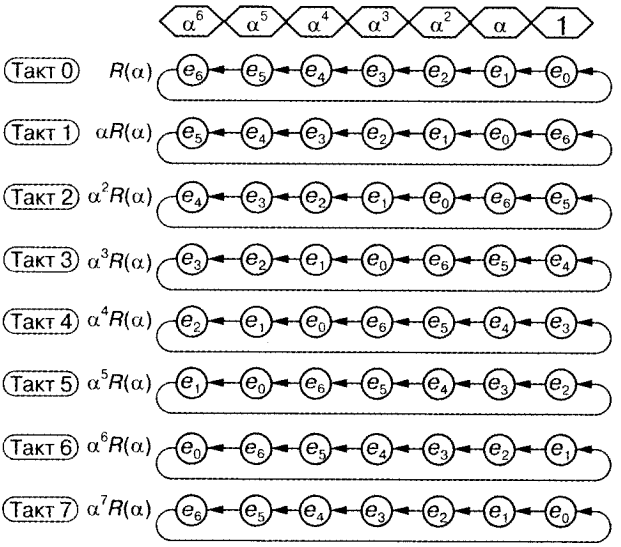


Рис. 13. Ланцюги конфігурацій помилок

У лінійних кодах за необхідності обчислення синдрому s помилка виправлялася додаванням за модулем 2 конфігурації помилки, яка відповідає цьому синдрому, до послідовності кодів, що приймаються. У циклічних кодах використовуючи циклічність залишку, що є синдромом, тобто циклічність конфігурації помилки, можна отримати конфігурацію синдрому (100) конфігурації помилки (0000001) і виправити помилку, додавши за модулем 2 конфігурацію цієї помилки до послідовності кодів, які приймаються. Проте в цьому випадку незалежно від того, додається чи ні "0" до конфігурації помилки, можна подати додавання "0", навіть, якщо до відповідного помилкового біта додається тільки "1".

У коді БЧХ (7, 4) показником переваг синдрому S є група з восьми величин:

- у разі безпомилкового синдрому — 0;
- у разі синдрому одиничної помилки — α^6, α^5, α^4, α^3, α^2, α, 1.

У разі декодування із застосуванням принципів циклічних кодів використовуючи циклічність залишків групи з семи величин α^6, α^5, α^4, α^3, α^2, α, 1 синдрому одиничної помилки, виправляють помилку виявленої синдромної конфігурації (100) одного з цих синдромів: $S = 1$. Отже, при декодуванні коду БЧХ (7, 4) синдром, який отримують для одиничної помилки, становить:

- у разі лінійних кодів $S = \alpha^6, \alpha^5, \alpha^4, \alpha^3, \alpha^2, \alpha, 1$;
- у разі циклічних кодів $S = 1$.

Тоді у разі декодування із застосуванням принципів циклічних кодів конфігурація детектора — це 1/7 конфігурації у разі декодування із застосуванням принципів лінійних кодів.

Оскільки у наведеному випадку, що стосується стандартного масиву коду (7, 4) (табл. 4), конфігурації синдрому і помилки співвідносяться як 1:1, то тут, замість того, щоб виявляти сім помилкових конфігурацій для виправлення всіх одиничних помилок при декодуванні лінійних кодів, виявляють тільки одну конфігурацію помилки при декодуванні циклічних кодів, тобто

Конфігурація виявлення помилок при декодуванні лінійних кодів	*Конфігурація виявлення помилок при декодуванні циклічних кодів*
(1000000) (0100000) ... (0000001) } сім конфігурацій	(0000001) одна конфігурація

Таким чином, порівнюючи методи декодування циклічних і лінійних кодів, одержуємо незначну перевагу числа отримуваних конфігурацій помилок, які є типовими основами.

17.2. БУФЕРНИЙ РЕГІСТР РЕКУРСИВНОГО ПРИЙОМУ

Після визначення залишку стає більш зрозумілим функціонування схеми ділення на породжувальний многочлен за принципом множення на початкову основу α. Зауважимо, що при декодуванні коду БЧХ (7, 4) із застосуванням принципів циклічних кодів використовується ділення на породжувальний многочлен за способом пошуку конфігурації помилки в циклічних кодах. Знаходиться конфігурація певної помилки, яка виправляється додаванням "1" за модулем 2 з бітом відповідної помилки.

З огляду на те, що результат змінюється не від того, додається чи ні "0" до конфігурації помилки, а лише від додавання "1" за модулем 2 до біта помилки, доцільно розглянути процес виправлення помилки у разі додавання за модулем 2 цієї конфігурації помилки до послідовності кодів, що приймаються.

На рис. 14 продемонстровано роботу приймального буферного регістра схеми декодування коду БЧХ (7, 4), послідовність виявлення конфігурації помилки (0000001) у часі та виведення сигналу виправлення помилки.

Як бачимо з рис. 14, конфігурація помилок, що виправляється, є конфігурацією помилки в сьомому біті, тобто конфігурацією помилки (0000001), якщо вона з'являється в біті останнього введення.

Схема ділення на породжувальний многочлен після визначення залишку $R(\alpha)$, як показано вище, діє за принципом множення на початкову основу α. Якщо продовжується робота схеми вигляду

$$R(\alpha) \to \alpha R(\alpha) \to \alpha^2 R(\alpha) \to \ldots \to$$
$$\to \alpha^7 R(\alpha)(=R(\alpha)) \to R(\alpha) \to \alpha R(\alpha) \to \ldots ,$$

то це вказує, скільки разів з'являється один і той самий залишок $R(\alpha)$.

Отже (див. рис. 14), всі одиничні помилки можна виявити тільки за конфігурацією помилки у сьомому біті. Далі, у разі виявлення помилки за цією конфігурацією важливо визначити, чи придатна конфігурація помилки в сьомому біті також для додавання за модулем 2 до послідовності кодів, що приймаються.

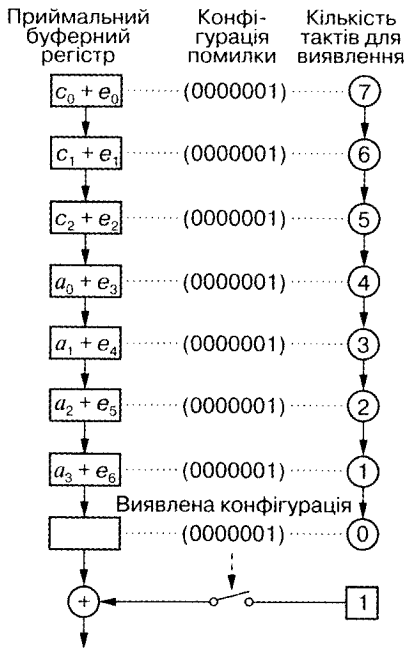


Рис. 14. Схема порядку виявлення конфігурації помилки

Оскільки код БЧХ (7, 4) — циклічний код, то його кодовий многочлен

$$v(x) = \beta_6 x^6 + \beta_5 x^5 + \beta_4 x^4 + \beta_3 x^3 + \beta_2 x^2 + \beta_1 x + x_0$$

є модулем $(x^7 + 1)$, який дорівнює "0", тому $x^7 + 1$. У цьому випадку, як це показано на рис. 15, якщо кодовий многочлен помножити на x, поміщений в регістр зсуву, то кожного разу, коли виконується черговий такт, тобто

$$xv(x) = b_5 x^6 + b_4 x^5 + b_3 x^4 + b_2 x^3 + b_1 x^2 + b_0 x + b_6,$$
$$x^2 v(x) = b_4 x^6 + b_3 x^5 + b_2 x^4 + b_1 x^3 + b_0 x^2 + b_6 x + b_5,$$
$$x^3 v(x) = b_3 x^6 + b_2 x^5 + b_1 x^4 + b_0 x^3 + b_6 x^2 + b_5 x + b_4,$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

кожен коефіцієнт біля x^6 (у найвищому степені), помножений на x, стає коефіцієнтом біля x^0 (у найнижчому степені), що дорівнює "1", а це означає циклічну дію. Таким чином, оскільки за такого циклу знову утворюється кодове слово, то код, що використовує породжувальний многочлен, названо циклічним.

Наведена точка зору на операцію циклічної підстановки у разі циклічних кодів надзвичайно важлива унаслідок додавання конфігурації помилки до послідовності кодів, що приймаються.

Враховуючи, що виправлення помилки здійснюється у разі виявлення помилки e_0 у сьомому біті, а залишок цього біта визначається в момент, коли послідовність кодів, що приймаються, повністю введена, розглянемо можливість циклічного переходу в подібну структуру послідовностей кодів, що приймаються. Подібна структура приймального буферного регістра наведена на рис. 16.

Як бачимо з цього рисунка, якщо приймальний буферний регістр перетвориться в циклічний, то, оскільки в такому циклічному приймальному буфер-

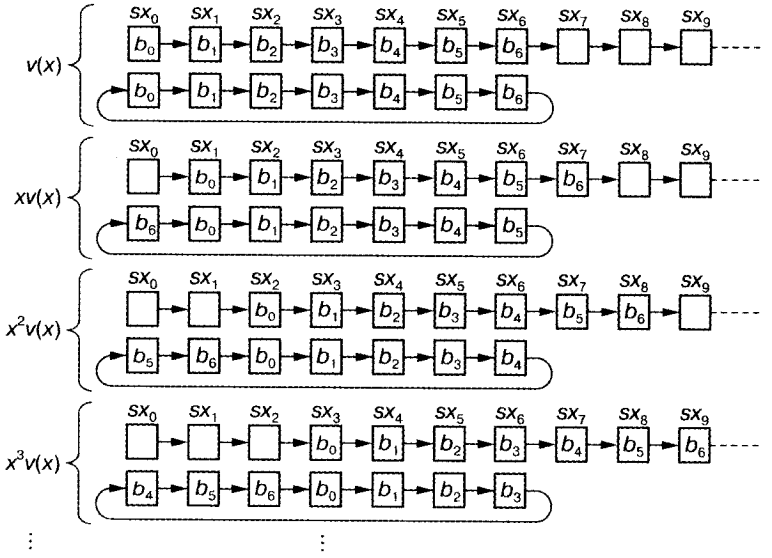


Рис. 15. Схема циклічної підстановки кодового слова циклічного коду

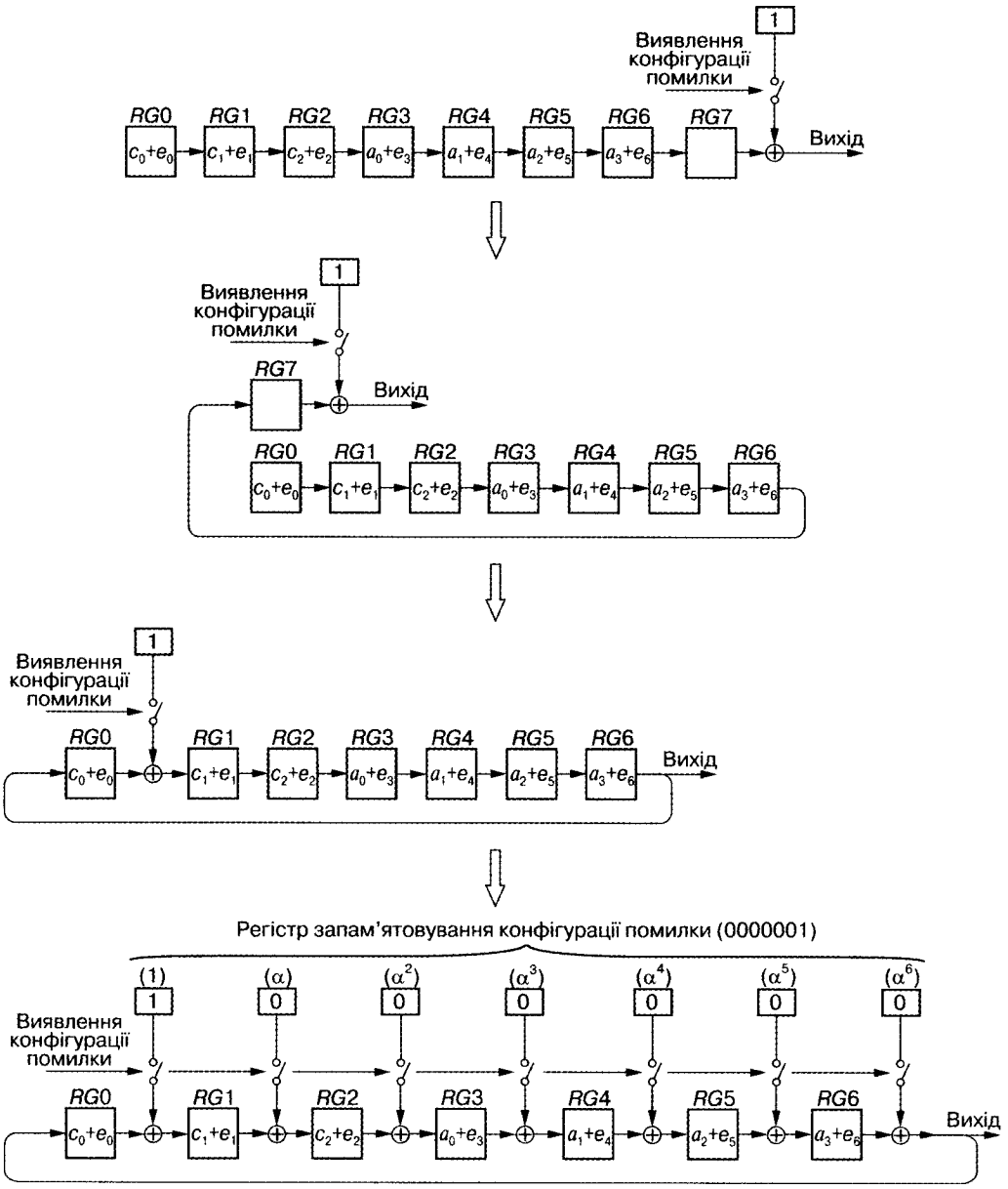


Рис. 16. Приймальний буферний регістр циклічного типу

ному регістрі зазвичай накопичується послідовність кодів, що приймаються, при виявленні помилки до послідовності кодів, які приймаються, додається конфігурація цієї помилки.

У схемах ділення на породжувальний многочлен у момент завершення повного введення послідовності кодів, що приймаються, знайдено залишок

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0$$

і в цей самий момент у приймальному буферному регістрі циклічного типу подібно, як і в

$$u(x) = (a_3 + e_6)x^6 + (a_2 + e_5)x^5 + (a_1 + e_4)x^4 + (a_0 + e_3)x^3 + \\ + (c_2 + e_2)x^2 + (c_1 + e_1)x + (c_0 + e_0),$$

накопичується послідовність кодів, що приймаються. Якщо схема запускається з цього моменту, то, як тільки утвориться залишок

$$R(\alpha) \to \alpha R(\alpha) \to \alpha^2 R(\alpha) \to \ldots \to \alpha^7 R(\alpha) = R(\alpha),$$

конфігурація помилки здійснить цикл, і, як тільки утвориться послідовність кодів приймального буферного регістра циклічного типу, що приймаються: $u(x) \to xu(x) \to x^2u(x) \to \ldots \to x^7u(x) = u(x)$, то повний цикл здійснить уже послідовність кодів, що приймаються.

Коли конфігурація помилок і послідовність кодів, що приймаються, здійснюють один цикл відповідно до процедури, наведеної у табл. 8, то обидві послідовності масивів координат бітів стають абсолютно однаковими.

Оскільки конфігурації помилок і кодової послідовності, що приймається, здійснюють цикли за однаковий час, то за час виявлення конфігурації синдрому можна (на підставі конфігурації синдрому) виправити помилку додаванням за модулем 2 конфігурації помилок до кодової послідовності, що приймається. У цьому і полягає спосіб декодування за дистанційним принципом.

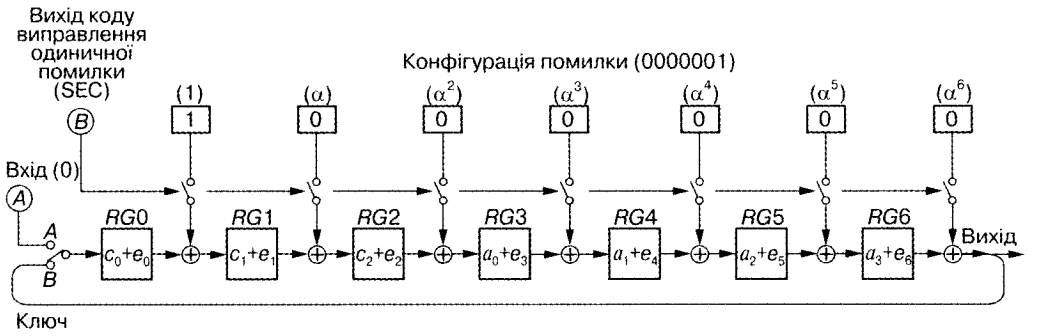
На рис. 17 подано схему декодування і виправлення помилок коду БЧХ (7, 4) за дистанційним принципом на базі буферного регістра циклічного типу. Проаналізувавши цей рисунок, одержимо, що незалежно від того, додається чи ні "0" конфігурації помилки до інформаційної послідовності, результат залишається однаковим, для спрощення схеми "0" можна не додавати. Оскільки у разі виявлення конфігурації синдрому вихід коду з виправленням одиничної помилки дорівнюватиме "1", то його можна використовувати замість "1" конфігурації помилки.

На рис. 18 зображено схему декодування за дистанційним принципом коду БЧХ (7, 4), в якій при виявленні помилки саме і використовується вихід коду з виправленням одиничної помилки.

Таблиця 8. Циклічність конфігурацій помилок і послідовності кодів, що приймаються

Такт	Конфігурація помилки в схемі ділення на породжувальний многочлен	Послідовність кодів приймального буферного регістра циклічного типу, що приймаються
0	$R(\alpha) : (e_6, e_5, e_4, e_3, e_2, e_1, e_0)$	$u(x) : (a_3 + e_6, a_2 + e_5, a_1 + e_4, a_0 + e_3, c_2 + e_2, c_1 + e_1, c_0 + e_0)$
1	$\alpha R(\alpha) : (e_5, e_4, e_3, e_2, e_1, e_0, e_6)$	$xu(x) : (a_2 + e_5, a_1 + e_4, a_0 + e_3, c_2 + e_2, c_1 + e_1, c_0 + e_0, a_3 + e_6)$
2	$\alpha^2 R(\alpha) : (e_4, e_3, e_2, e_1, e_0, e_6, e_5)$	$x^2u(x) : (a_1 + e_4, a_0 + e_3, c_2 + e_2, c_1 + e_1, c_0 + e_0, a_3 + e_6, a_2 + e_5)$
3	$\alpha^3 R(\alpha) : (e_3, e_2, e_1, e_0, e_6, e_5, e_4)$	$x^3u(x) : (a_0 + e_3, c_2 + e_2, c_1 + e_1, c_0 + e_0, a_3 + e_6, a_2 + e_5, a_1 + e_4)$
4	$\alpha^4 R(\alpha) : (e_2, e_1, e_0, e_6, e_5 e_4, e_3)$	$x^4u(x) : (c_2 + e_2, c_1 + e_1, c_0 + e_0, a_3 + e_6, a_2 + e_5, a_1 + e_4, a_0 + e_3)$
5	$\alpha^5 R(\alpha) : (e_1, e_0, e_6, e_5, e_4, e_3, e_2)$	$x^5u(x) : (c_1 + e_1, c_0 + e_0, a_3 + e_6, a_2 + e_5, a_1 + e_4, a_0 + e_3, c_2 + e_2)$
6	$\alpha^6 R(\alpha) : (e_0, e_6, e_5, e_4, e_3, e_2, e_1)$	$x^6u(x) : (c_0 + e_0, a_3 + e_6, a_2 + e_5, a_1 + e_4, a_0 + e_3, c_2 + e_2, c_1 + e_1)$
7	$\alpha^7 R(\alpha) : (e_6, e_5, e_4, e_3, e_2, e_1, e_0)$	$x^7u(x) : (a_3 + e_6, a_2 + e_5, a_1 + e_4, a_0 + e_3, c_2 + e_2, c_1 + e_1, c_0 + e_0)$

Декодування циклічних кодів за дистанційним принципом



Позиція ключа	Такт	"1" у разі конфігурації (100) виходу коду виправлення одиничної помилки (SEC)	Приймальний буферний регістр циклічного типу							Вихід декодування	Конфігурація виявлених і виправлених помилок
			*RG*0	*RG*1	*RG*2	*RG*3	*RG*4	*RG*5	*RG*6		
B	0	e_0	$c_0 + e_0$	$c_1 + e_1$	$c_2 + e_2$	$a_0 + e_3$	$a_1 + e_4$	$a_2 + e_5$	$a_3 + e_6$	—	(000000 e_0)
	1	e_6	$a_3 + e_6$	c_0	$c_1 + e_1$	$c_2 + e_2$	$a_0 + e_3$	$a_1 + e_4$	$a_2 + e_5$	—	(000000 e_6)
	2	e_5	$a_2 + e_5$	a_3	c_0	$c_1 + e_1$	$c_2 + e_2$	$a_0 + e_3$	$a_1 + e_4$	—	(000000 e_5)
	3	e_4	$a_1 + e_4$	a_2	a_3	c_0	$c_1 + e_1$	$c_2 + e_2$	$a_0 + e_3$	—	(000000 e_4)
	4	e_3	$a_0 + e_3$	a_1	a_2	a_3	c_0	$c_1 + e_1$	$c_2 + e_2$	—	(000000 e_3)
	5	e_2	$c_2 + e_2$	a_0	a_1	a_2	a_3	c_0	$c_1 + e_1$	—	(000000 e_2)
	6	e_1	$c_1 + e_1$	c_2	a_0	a_1	a_2	a_3	c_0	—	(000000 e_1)
A	7	—	c_0	c_1	c_2	a_0	a_1	a_2	a_3	a_3	—
	8	—	0	c_0	c_1	c_2	a_0	a_1	a_2	a_2	—
	9	—	0	0	c_0	c_1	c_2	a_0	a_1	a_1	—
	10	—	0	0	0	c_0	c_1	c_2	a_0	a_0	—
	11	—	0	0	0	0	c_0	c_1	c_2	c_2	—
	12	—	0	0	0	0	0	c_0	c_1	c_1	—
	13	—	0	0	0	0	0	0	c_0	c_0	—

У послідовності e_6—e_0 лише один біт дорівнює "1"

Рис. 17. Схема декодування і виправлення помилок коду БЧХ (7, 4) за дистанційним принципом

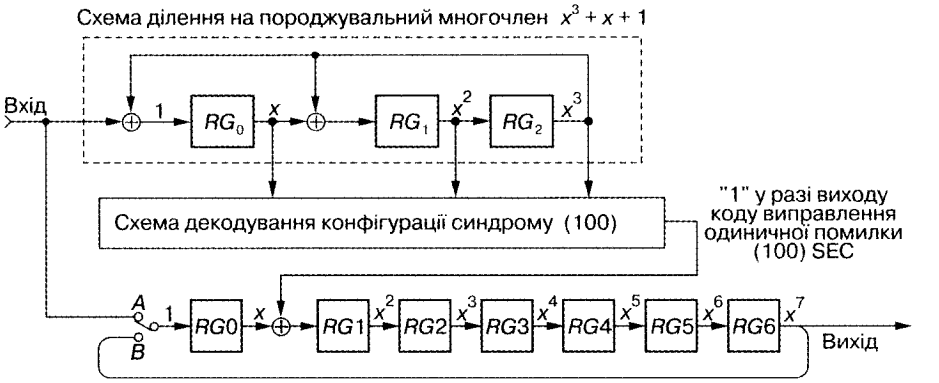


Рис. 18. Схема декодування коду БЧХ (7, 4), що використовує вихід коду з виправленням одиничної помилки

Порівнявши цю схему з попередньою схемою декодування коду БЧХ (7, 4) (див. табл. 8), бачимо, що схема декодування за дистанційним принципом (див. рис. 18) значно складніша. Отже, в розглянутому випадку коду з виправленням одиничної помилки використання попередньої схеми декодування є зручнішим.

Спосіб декодування за дистанційним принципом є результативним, коли коди БЧХ мають розмірність більше 10.

У разі конфігурації одиничної помилки декодування за дистанційним принципом є необов'язковим. Проте він стає необхідним у разі виправлення конфігурації помилок, що містить більше ніж дві помилки.

17.3. МОДЕЛІ ВІДСТАНІ ПОМИЛКОВИХ БІТІВ

Нехай α є коренем простого многочлена $x^3 + x + 1$, тоді залишок $R(\alpha)$ від ділення послідовності помилок $e = (e_6, e_5, e_4, e_3, e_2, e_1, e_0)$ в коді БЧХ (7, 4) на характеристичний многочлен $x^3 + x + 1$ визначається формулою (7), тобто

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0,$$

і цей запис є загальним поданням залишку для семибітового БЧХ коду, що формується за допомогою кореня α. Інакше кажучи, яким би не був породжувальний многочлен, залишок можна виразити так, як його подано у формулі (7).

Якщо $G(x)$ — породжувальний многочлен, то, оскільки $G(\alpha) = 0$, дійсно формула (7) має менший порядок, ніж порядок породжувального многочлена.

Завершивши обчислення залишку $R(\alpha)$ за схемою ділення породжувального многочлена, далі у разі продовження ділення за цією схемою використовуємо її як схему обчислення початкового елемента α. Помноживши залишок $R(\alpha)$ на початковий елемент α, отримаємо циклічну послідовність помилок (див. рис. 12). У разі циклічної послідовності помилок порядок проходження помилкових бітів не змінюється.

Наприклад, встановимо в "1" сьомий і шостий біти в послідовності помилок e. Тоді послідовність помилок має вигляд (0000011), тобто $e_1 = 1$, $e_0 = 1$, а решта всіх бітів дорівнює нулю.

У тому випадку, коли послідовність помилок циклічна, порядок проходження e_1 і e_0 не відрізняється від наступного:

$$\begin{array}{ccccccc}
(0 & 0 & 0 & 0 & 0 & e_1 & e_0) \\
(0 & 0 & 0 & 0 & e_1 & e_0 & 0) \\
(0 & 0 & 0 & e_1 & e_0 & 0 & 0) \\
(0 & 0 & e_1 & e_0 & 0 & 0 & 0) \\
(0 & e_1 & e_0 & 0 & 0 & 0 & 0) \\
(e_1 & e_0 & 0 & 0 & 0 & 0 & 0) \\
(e_0 & 0 & 0 & 0 & 0 & 0 & e_1) \\
(0 & 0 & 0 & 0 & 0 & e_1 & e_0)
\end{array}$$

Для спрощеного подання послідовності помилок використовуватимемо тільки біти, що дорівнюють "1". Наприклад, послідовність помилок, що має

одиниці в шостому і сьомому бітах, записується так: (e_1, e_0). Виразивши послідовність помилок комбінацією помилкових бітів, отримаємо наступну відповідність:

Модель помилок (e_1, e_0) $\leftrightarrow$ *Послідовність помилок* (0000011)

У разі використання такого способу позначення послідовність помилок (0001011) відповідатиме моделі помилок (e_3, e_1, e_0). У тому випадку, коли в послідовності помилок відсутні одиничні біти, модель помилок має вигляд "0".

Модель помилок відповідає циклічній схемі ділення породжувального многочлена, тому її можна подати у формі кола (рис.19).

Проаналізуємо відстані між помилковими бітами в моделі помилок, зображеній у формі кола. Тоді відстані можна обчислити, знаючи необхідне число тактів, потрібних для досягнення наступного помилкового біта. Якщо α множити деяке число разів, то можна подати положення наступних помилкових бітів. У найпростішому випадку для моделі помилок, що має форму кола, помилковий біт і відстані між помилковими бітами визначаються рахунком відстаней між заданими бітами у разі руху по колу.

Наприклад, у разі моделі помилки (e_1, e_0), яка наведена на рис. 20, відстань від e_0 до e_1 дорівнює одиниці, а від e_1 до e_0 — шести.

У разі одиничної помилки помилковий біт тільки один. Тоді для моделі помилок (e_0) відстань між помилковими бітами e_0 і e_0 дорівнює семи (рис. 21).

У разі трьох помилок, наприклад, (e_3, e_1, e_0) відстань між e_0 і e_1 дорівнює одиниці, відстань між e_1 і e_3 — двом, а відстань між e_3 і e_0 — чотирьом (рис. 22).

Розглянемо відстань між помилковими бітами у разі, коли залишок $R(\alpha)$ циклічний унаслідок ділення породжувального многочлена.

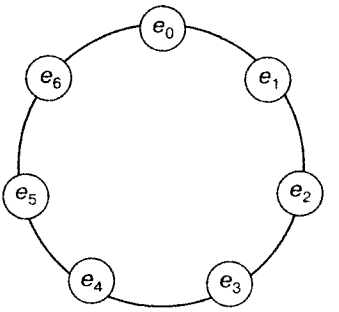


Рис. 19. Модель помилок

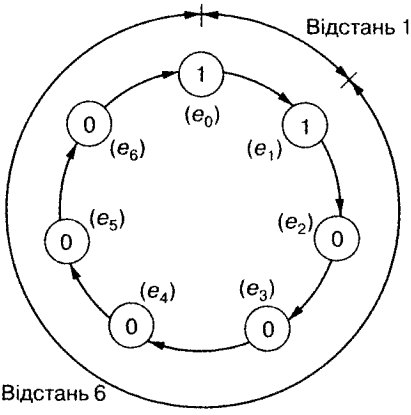


Рис. 20. Схема для визначення відстані для моделі помилок (e_1, e_0)

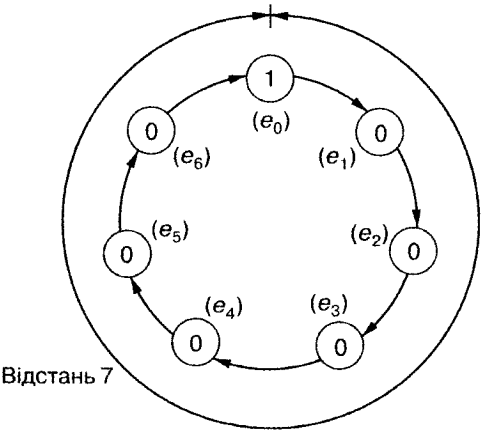


Рис. 21. Схема для визначення відстані для моделі помилки (e_0)

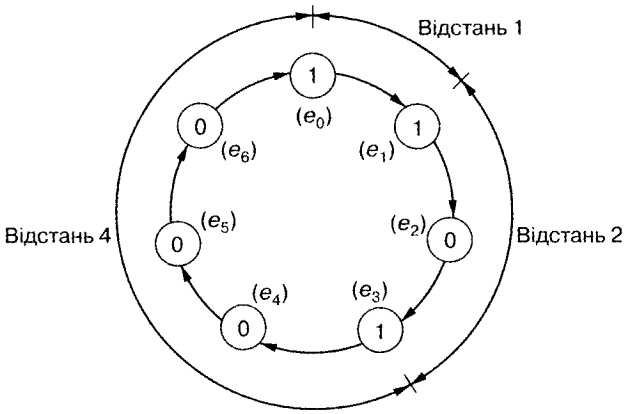


Рис. 22. Схема для визначення відстані для моделі помилок (e_3, e_1, e_0)

Наприклад, при помилках в шостому і сьомому бітах модель помилок має вигляд (e_1, e_0). Тоді залишок набуває вигляду $R(\alpha) = \alpha + 1$. Відобразимо його в циклічній формі за допомогою схеми ділення породжувального многочлена. Для цього залишку початковий елемент α є множником і $\alpha^7 = 1$, тому

$$\alpha R(\alpha) = \alpha^2 + \alpha \leftrightarrow \text{модель помилок } (e_2, e_1),$$

$$\alpha^2 R(\alpha) = \alpha^3 + \alpha^2 \leftrightarrow \text{модель помилок } (e_3, e_2),$$

$$\alpha^3 R(\alpha) = \alpha^4 + \alpha^3 \leftrightarrow \text{модель помилок } (e_4, e_3),$$

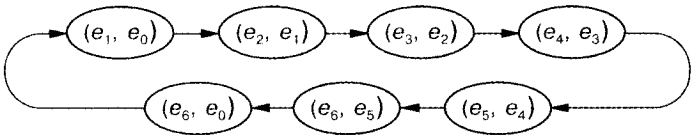
$$\alpha^4 R(\alpha) = \alpha^5 + \alpha^4 \leftrightarrow \text{модель помилок } (e_5, e_4),$$

$$\alpha^5 R(\alpha) = \alpha^6 + \alpha^5 \leftrightarrow \text{модель помилок } (e_6, e_5),$$

$$\alpha^6 R(\alpha) = 1 + \alpha^6 \leftrightarrow \text{модель помилок } (e_7, e_6),$$

$$\alpha^7 R(\alpha) = \alpha + 1 \leftrightarrow \text{модель помилок } (e_1, e_0).$$

Зазначимо, що для моделі помилок (e_1, e_0) є дві відстані: 1 і 6. Зрозуміло, що структура відстані (1 або 6) для моделі помилок (e_1, e_0) протягом одного циклу не змінюється, тобто



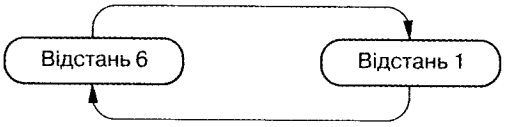
Очевидно, щоб визначити залишок, що має циклічне подання, необхідно однаковим чином множити залишок на початковий елемент α. Тому модель помилок, яка має описану структуру відстані, є однією моделлю синдрому.

Доцільно тепер розробити модель відстані помилкових бітів. Відстань між помилковими бітами для моделі помилок (e_1, e_0) буде такою: (відстань 1, відстань 6). Уклавши послідовно два значення в дужки, можна отримати просту форму зображення моделі відстані для помилкових бітів. Для розглянутої моделі помилок — це модель відстані вигляду (1, 6). Сім моделей помилок, які мають таку саму модель відстані, можна подати так:

$$(e_1, e_0), (e_2, e_1), (e_3, e_2), (e_4, e_3), (e_5, e_4), (e_6, e_5), (e_6, e_0).$$

Модель відстані (1, 6) і модель відстані (6, 1) однакові.

Розглянемо схему, що демонструє процедуру, за якою відбувається циклічна підстановка для моделі відстані (6, 1).



Розглянемо спочатку мінімальну відстань. У разі одиничної помилки модель відстані (див. рис. 21) є такою моделлю відстані (7), за допомогою якої можна виявити всі одиничні помилки. На рис. 22 наведено модель відстані для моделі трьох помилок (e_3, e_1, e_0), тут модель відстані має вигляд (1, 2, 4).

Послідовність чисел, що знаходяться у круглих дужках, для моделі відстані, зображеної таким чином, вказує бітове число моделі помилок, а саме: кодове бітове число, а також кількість помилок, що відповідно наведено нижче:

Модель відстані (7)	Кодове бітове число дорівнює семи	Одинична помилка
Модель відстані (1, 6)	Кодове бітове число дорівнює семи	Подвійна помилка
Модель відстані (1, 2, 4)	Кодове бітове число дорівнює семи	Потрійна помилка

Наприклад, якщо взяти точно таку саму модель помилок для моделі відстані, то для неї можна за допомогою однієї конфігурації синдрому виявити цю помилку. Тому розглянемо модель відстані і конфігурацію синдрому, тобто відношення залишку у разі коду БЧХ з кодовим бітовим числом 7.

Нехай первинний многочлен — $x^3 + x + 1$. Його коренем є α. У цьому випадку залишок $R(\alpha)$ породжувального многочлена, як було показано вище, записується так:

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0.$$

Обчислимо залишок $R(\alpha)$ у момент закінчення введення вхідної кодової послідовності. Цей залишок після виконання операції ділення є моделлю відстані помилок.

Обчислюючи залишок як конфігурації синдрому з використанням формули (7), за початкову точку приймемо e_0. Дійсно, α є коренем породжувального многочлена $G(x)$, тому залишок $R(\alpha)$ буде многочленом меншого порядку, ніж породжувальний многочлен. Отже, бітове число синдрому є порядком m породжувального многочлена.

Кодове бітове число дорівнює 7, тому можливими є не більше ніж три типи моделей відстані помилок.

17.3.1. Моделі відстані одиничної помилки

У цьому випадку модель відстані має вигляд (7), а залишок — $R(\alpha) = \alpha^0 = 1$. Конфігурацією синдрому, що показує цей залишок, можливе виявлення моделі одиничної помилки з моделлю відстані (7) для

$$(e_0), (e_1), (e_2), (e_3), (e_4), (e_5), (e_6).$$

Число моделей одиничної помилки в цілому дорівнює 7:

$$\binom{7}{1} = \frac{7!}{1!(7-1)!} = \frac{7!}{1!6!} = 7.$$

Отже, цією моделлю відстані можна виявити всі моделі помилок.

17.3.2. Моделі відстані подвійної помилки

Якщо розглянути комбінацію семибітового коду, то модель відстані подвійної помилки можна просто подати у вигляді двох цифр, що записані в дужках. У цьому разі можливі такі три пари моделей відстані: (1, 6), (2, 5), (3, 4). Зауважимо, що моделі відстані (4, 3) і відстані (3, 4) однакові.

Модель відстані для подвійних помилок можна подати так: (i, j), де $i \le j$, $i + j = n$. У цьому випадку залишок для моделі записується таким виразом:

$$R(\alpha) = \alpha^1 + \alpha^0 = \alpha^1 + 1.$$

Тоді у разі коду з кодовим бітовим числом, яке дорівнює 7, залишок для даних трьох моделей відстані набуде відповідно вигляду:

- для моделі відстані (1, 6) — $R(\alpha) = \alpha^1 + 1$;
- для моделі відстані (2, 5) — $R(\alpha) = \alpha^2 + 1$;
- для моделі відстані (3, 4) — $R(\alpha) = \alpha^3 + 1$.

Використовуючи модель відстані (1, 6), можна виявити сім моделей помилок: (e_1, e_0), (e_2, e_1), (e_3, e_2), (e_4, e_3), (e_5, e_4), (e_6, e_5), (e_6, e_0).

Залишком моделі відстані (2, 5) є многочлен вигляду: $R(\alpha) = \alpha^2 + 1$. Тому, множачи послідовно протягом одного циклу залишок $R(\alpha)$ на корінь α, можна отримати такі моделі помилок:

$$R(\alpha) = \alpha^2 + 1 \leftrightarrow \text{модель помилки } (e_2, e_0),$$
$$\alpha R(\alpha) = \alpha^3 + \alpha \leftrightarrow \text{модель помилки } (e_3, e_1),$$
$$\alpha^2 R(\alpha) = \alpha^4 + \alpha^2 \leftrightarrow \text{модель помилки } (e_4, e_2),$$
$$\alpha^3 R(\alpha) = \alpha^5 + \alpha^3 \leftrightarrow \text{модель помилки } (e_5, e_3),$$
$$\alpha^4 R(\alpha) = \alpha^6 + \alpha^4 \leftrightarrow \text{модель помилки } (e_6, e_4),$$
$$\alpha^5 R(\alpha) = \alpha^5 + 1 \leftrightarrow \text{модель помилки } (e_5, e_0),$$
$$\alpha^6 R(\alpha) = \alpha^6 + \alpha \leftrightarrow \text{модель помилки } (e_6, e_1),$$
$$\alpha^7 R(\alpha) = R(\alpha) \leftrightarrow \text{модель помилки } (e_2, e_0).$$

Тобто, використовуючи модель відстані (2, 5) можна виявити сім таких моделей помилок: (e_2, e_0), (e_3, e_1), (e_4, e_2), (e_5, e_3), (e_6, e_4), (e_5, e_0), (e_6, e_1).

Аналогічно, за допомогою моделі відстані (3, 4) можна виявити сім наступних моделей помилок: (e_3, e_0), (e_4, e_1), (e_5, e_2), (e_6, e_3), (e_4, e_0), (e_5, e_1), (e_6, e_2).

Отже, загальне число моделей помилок, які можна виявити трьома вказаними моделями відстаней, становить величину $7 \cdot 3 = 21$. Загальне число моделей з подвійною помилкою при цьому визначається так:

$$\binom{7}{2} = \frac{7!}{2!(7-2)!} = \frac{7!}{2!5!} = \frac{6 \cdot 7}{2} = 21.$$

З наведеного вище маємо, що різні моделі подвійних помилок можна виявити за допомогою всього трьох моделей відстаней помилок.

17.3.3. Моделі відстані потрійної помилки

Якщо подати модель відстані для потрійних помилок у вигляді комбінації (i, j, k), то між елементами такої комбінації повинно виконуватися наступне співвідношення: $i + j + k = 7$, де $i \leq j, k$.

Оскільки модель відстані, наприклад, для циклічної комбінації (1, 5, 1) має наступний зв'язок:

то, використовуючи циклічну перестановку (1, 5, 1) → (1, 1, 5), отримуємо модель відстані (1, 1, 5).

Якщо модель відстані комбінації (1, 2, 4) має зв'язок

то нова модель відстані — (1, 4, 2).

Якщо обчислити так само моделі відстаней для потрійних помилок, то отримаємо п'ять таких моделей:

- модель відстані (1, 1, 5);
- модель відстані (1, 2, 4);
- модель відстані (1, 3, 3);
- модель відстані (1, 4, 2);
- модель відстані (2, 2, 3).

На підставі визначення моделі відстані (i, j, k) і того, що

$$R(\alpha) = \alpha^{i+j} + \alpha^{i} + \alpha^{0} = \alpha^{i+j} + 1,$$

неважко обчислити відповідні залишки для різних моделей відстаней. Взаємозв'язок між моделями відстаней і відповідними їм залишками подано нижче у вигляді таблиці відповідностей.

Модель відстані		Залишок
(1, 1, 5)	↔	$R(\alpha) = \alpha^2 + \alpha + 1$
(1, 2, 4)	↔	$R(\alpha) = \alpha^3 + \alpha + 1$
(1, 3, 3)	↔	$R(\alpha) = \alpha^4 + \alpha + 1$
(1, 4, 2)	↔	$R(\alpha) = \alpha^5 + \alpha + 1$
(2, 2, 3)	↔	$R(\alpha) = \alpha^4 + \alpha^2 + 1$

Для того, щоб визначити число моделей помилок за допомогою різних моделей відстаней, слід помножити залишок $R(\alpha)$ на α, продовжуючи це множення до виявлення повторення елементів моделі.

Для прикладу виберемо модель відстані (1, 1, 5) і виявимо, які моделі помилок будуть виникати. Виконавши запропоновану процедуру, отримаємо сім моделей помилок:

$$R(\alpha) = \alpha^2 + \alpha + 1 \leftrightarrow \text{модель помилки } (e_2, e_1, e_0),$$
$$\alpha R(\alpha) = \alpha^3 + \alpha^2 + \alpha \leftrightarrow \text{модель помилки } (e_3, e_2, e_1),$$
$$\alpha^2 R(\alpha) = \alpha^4 + \alpha^3 + \alpha^2 \leftrightarrow \text{модель помилки } (e_4, e_3, e_2),$$
$$\alpha^3 R(\alpha) = \alpha^5 + \alpha^4 + \alpha^3 \leftrightarrow \text{модель помилки } (e_5, e_4, e_3),$$
$$\alpha^4 R(\alpha) = \alpha^6 + \alpha^5 + \alpha^4 \leftrightarrow \text{модель помилки } (e_6, e_5, e_4),$$
$$\alpha^5 R(\alpha) = \alpha^6 + \alpha^5 + 1 \leftrightarrow \text{модель помилки } (e_6, e_5, e_0),$$
$$\alpha^6 R(\alpha) = \alpha^6 + \alpha + 1 \leftrightarrow \text{модель помилки } (e_6, e_1, e_0).$$
$$\alpha^7 R(\alpha) = R(\alpha) \leftrightarrow \text{модель помилки } (e_2, e_1, e_0).$$

Тобто всі сім виявлених моделей помилок визначаються однією моделлю відстані (1, 1, 5).

Аналогічно цьому і чотири інші моделі відстаней для потрійних помилок, що залишилися, виявляють по сім моделей помилок кожна. Моделі помилок, які можна виявити, і моделі відстаней для потрійних (а також подвійних і одиничних) помилок зведені в табл. 9.

Однією моделлю відстані можна виявити сім моделей помилок, тому п'ятьма моделями відстані для потрійної помилки можна виявити $7 \cdot 5 = 35$ моделей потрійної помилки. Загальне число моделей з потрійною помилкою при цьому становить

$$\binom{7}{3} = \frac{7!}{3!(7-3)!} = \frac{7!}{3!4!} = \frac{7 \cdot 6 \cdot 5}{3 \cdot 2} = 35,$$

Таблиця 9. Моделі відстані помилок для моделей помилок, що містять до трьох помилок

Число помилкових бітів	Модель помилок	Можливі моделі помилок для моделі відстані помилок
Одинична помилка	(7)	(e_0), (e_1), (e_2), (e_3), (e_4), (e_5), (e_6)
Подвійна помилка	(1, 6) (2, 5) (3, 4)	(e_1, e_0), (e_2, e_1), (e_3, e_2), (e_4, e_3), (e_5, e_4), (e_6, e_5), (e_6, e_0) (e_2, e_0), (e_3, e_1), (e_4, e_2), (e_5, e_3), (e_6, e_4), (e_5, e_0), (e_6, e_1) (e_3, e_0), (e_4, e_1), (e_5, e_2), (e_6, e_3), (e_4, e_0), (e_5, e_1), (e_6, e_2)
Потрійна помилка	(1, 1, 5) (1, 2, 4) (1, 3, 3) (1, 4, 2) (2, 2, 3)	(e_2, e_1, e_0), (e_3, e_2, e_1), (e_4, e_2, e_2), (e_5, e_4, e_3), (e_6, e_5, e_4), (e_6, e_5, e_0), (e_6, e_1, e_0) (e_3, e_1, e_0), (e_4, e_2, e_1), (e_5, e_3, e_2), (e_6, e_4, e_3), (e_5, e_4, e_0), (e_6, e_5, e_1), (e_6, e_2, e_0) (e_4, e_1, e_0), (e_5, e_2, e_1), (e_6, e_3, e_2), (e_4, e_3, e_0), (e_5, e_4, e_1), (e_6, e_5, e_2), (e_6, e_3, e_0) (e_5, e_1, e_0), (e_6, e_2, e_1), (e_3, e_2, e_0), (e_4, e_3, e_1), (e_5, e_4, e_2), (e_6, e_5, e_3), (e_6, e_4, e_0) (e_4, e_2, e_0), (e_5, e_3, e_1), (e_6, e_4, e_2), (e_5, e_3, e_0), (e_6, e_4, e_1), (e_5, e_2, e_0), (e_6, e_3, e_1)

Таблиця 10. Конфігурація синдрому і модель відстані помилок для кодів БЧХ з кодовим бітовим числом, що дорівнює 7

Число помилкових бітів	Модель відстані помилок	Залишок моделі відстані помилок	Число корегованих моделей помилок	
Одинична помилка	(7)	$R(\alpha) = 1$	7	
Подвійна помилка	(1, 6)	$R(\alpha) = \alpha + 1$	7	21
	(2, 5)	$R(\alpha) = \alpha^2 + 1$	7	
	(3, 4)	$R(\alpha) = \alpha^3 + 1$	7	
Потрійна помилка	(1, 1, 5)	$R(\alpha) = \alpha^2 + \alpha + 1$	7	35
	(1, 2, 4)	$R(\alpha) = \alpha^3 + \alpha + 1$	7	
	(1, 3, 3)	$R(\alpha) = \alpha^4 + \alpha + 1$	7	
	(1, 4, 2)	$R(\alpha) = \alpha^5 + \alpha + 1$	7	
	(2, 2, 3)	$R(\alpha) = \alpha^4 + \alpha + 1$	7	

тому цими п'ятьма моделями відстані можна виявити всі моделі потрійної помилки.

У табл. 10 наведено відповідність залишку $R(\alpha)$ і моделі відстані помилки для всіх моделей помилок, аж до моделей потрійної помилки, для кодів БЧХ з бітовим кодовим числом, що дорівнює 7.

Якщо прийняти за циклічний код лінійний код, то створюючи стандартну конфігурацію, можна повністю корегувати модель помилок для даного коду. Серед моделей помилок, які можна корегувати в циклічному коді, корегування структури відстані помилкового біта відбувається так само. Тобто, очевидно, що за допомогою конфігурації одного синдрому можна виявити модель помилок, яка відповідає такій самій моделі відстані помилок.

Для розглянутої моделі відстані помилок зупинимося на тій моделі, яка має структуру відстані з початковою точкою e_0.

Розділ 18

ДЕКОДУВАННЯ ЗА МОДЕЛЛЮ ВІДСТАНІ ПОМИЛОК

Із співвідношення моделі помилок і моделі відстані помилок випливає такий порядок декодування за моделлю відстані помилок:

1. За допомогою схеми ділення породжувального многочлена формується циклічний залишок.

2. За допомогою циклічного залишку визначається конфігурація синдрому моделі відстані помилок.

3. У приймальному буферному регістрі циклічна модель відстані помилок додається до циклічного коду, що приймається.

4. Виконується виправлення помилок.

Тому для виконання декодування за моделлю відстані помилок потрібно чотири основні схеми:

1. Схема ділення многочлена, що породжує і має обчислюваний циклічний залишок.

2. Схема виявлення конфігурації синдрому в циклічному залишку.

3. Схема визначення моделі відстаней помилок, необхідна для визначення моделі відстані, відповідної виявленої конфігурації синдрому.

4. Схема перетворення кодової послідовності, що приймається, в циклічну форму і її синхронізація із залишком. Для виправлення помилок доцільно використовувати буферний регістр циклічного типу з суматорами за модулем 2.

Структурна схема декодування за моделлю відстані помилок наведена на рис. 23.

Розглянемо роботу схеми декодування при використанні простих кодів. Нехай α — корінь первинного многочлена $x^3 + x + 1$. З цього многочлена можна отримати код БЧХ з кодовим бітовим числом, яке дорівнює 7.

Коди БЧХ (7, 4) SEC є кодами, отриманими з породжувального многочлена $x^3 + x + 1$, коренями якого є степенева послідовність кореня α, тобто α, α^2. Коди БЧХ є кодами SEC, тому, якщо визначено модель відстані одиничної помилки, то можливим є декодування за моделлю відстані помилок.

Пояснимо схему декодування за моделлю відстані для коду БЧХ (7, 4), наведену на рис. 18. Як випливає з табл. 9, модель відстані одиничної помилки єдина — (7). За допомогою цієї єдиної моделі відстані можна виявити всі моделі одиничних помилок. Залишок цієї моделі відстані $R(\alpha) = 1$. Конфігурація синдрому, що вказує на цей залишок, має вигляд ($RG0$, $RG1$, $RG2$) = (1, 0, 0).

Вхідна кодова послідовність після буферного регістра стає циклічною і синхронізованою із залишком, що отримується в схемі ділення породжувального многочлена. Отже, при виявленні конфігурації синдрому вигляду (1, 0, 0) ви-

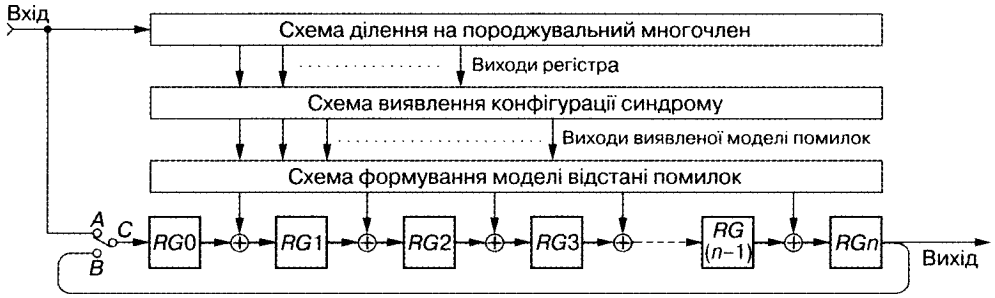


Рис. 23. Структурна схема декодування за моделлю відстані помилок для циклічних кодів і багатократних помилок

правлення помилок здійснюється шляхом додавання за модулем 2 моделі відстані помилок і кодової послідовності, що приймається.

Розглянемо тепер декодування за моделлю відстані, узявши як приклад код БЧХ з багатократною помилкою і кодовим бітовим числом, що дорівнює 7. Таким кодом БЧХ, утвореним із породжувального многочлена, з коренями $(\alpha, \alpha^2, ..., \alpha^6)$, є код (7, 1) ТЕС. Для цього коду маємо

- кодове бітове число $n = 7$;
- контрольне бітове число $m = 6$;
- інформаційне бітове число $k = 1$;
- можливість виправлення помилки ТЕС (виправлення потрійної помилки);
- породжувальний многочлен $x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$.

Зауважимо, що цей код не є застосовним на практиці, а є прикладом коду БЧХ. Принцип кодування і структурну схему кодування кодом БЧХ (7, 1) наведено на рис. 24.

Як бачимо з цього рисунка, кодове слово має вигляд

$$v = (a_0, a_0, a_0, a_0, a_0, a_0, a_0).$$

У разі зміни інформаційного біта a_0 змінюються всі біти, тому відстань між кодами дорівнює 7. Отже, код ТЕС може використовуватися для виправлення помилок. Для нього можливий вибір з таких ситуацій:

1. Модель без помилок.
2. Модель з однією помилкою.
3. Модель з двома помилками.
4. Модель з трьома помилками.

Оскільки для даного коду породжувальний многочлен має порядок, що дорівнює шести, то контрольне бітове число теж дорівнює 6, а число контрольованих ситуацій, яке можна собі уявити, $2^6 = 64$.

Можна підрахувати загальне число моделей помилок, що включають в себе:

- число моделей без помилок — $\binom{7}{0} = 1$;
- число моделей з одиничною помилкою — $\binom{7}{1} = 7$;

1 Інформаційна послідовність a → Інформаційний многочлен $A(x)$

$a—a$ $A(x) = a_0$

2 Породжувальний многочлен $G(x)$ → Порядок x^6 породжувального многочлена, помножений на $A(x)$

$G(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$ $A(x)x^6 - a_0x^6$

3 Залишок $R(x)$

$$(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1) \quad \frac{a_0}{a_0x^6}$$

$A(x)x^6/G(x)$

$$\frac{a_0x^6 + a_0x^5 + a_0x^4 + a_0x^3 + a_0x^2 + a_0x + a_0}{a_0x^5 + a_0x^4 + a_0x^3 + a_0x^2 + a_0x + a_0}$$

→ $R(x) = a_0x^5 + a_0x^4 + a_0x^3 + a_0x^2 + a_0x + a_0$

4 $A(x)x^6 + R(x)$ → Кодований многочлен $v(x)$

$v(x) = a_0x^6 + a_0x^5 + a_0x^4 + a_0x^3 + a_0x^2 + a_0x + a_0$

5 Кодований многочлен $v(x)$ → Послідовність кодів передач v

$v = (a_0, a_0, a_0, a_0, a_0, a_0, a_0)$

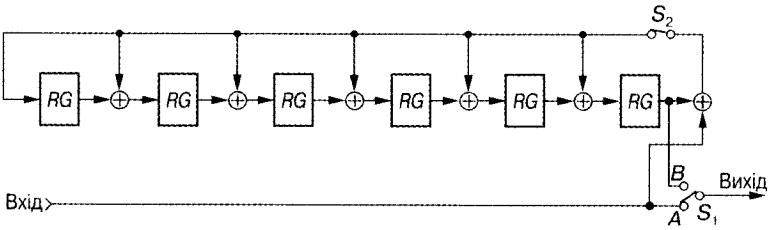


Рис. 24. Структурна схема і порядок кодування коду БЧХ (7, 1)

- число моделей з подвійною помилкою — $\left(\frac{7}{2}\right) = 21$;
- число моделей з потрійною помилкою — $\left(\frac{7}{3}\right) = 35$.

Отож, загальна кількість моделей помилок визначається так:

$$\left(\frac{7}{0}\right) + \left(\frac{7}{1}\right) + \left(\frac{7}{2}\right) + \left(\frac{7}{3}\right) = 1 + 7 + 21 + 35 = 64.$$

Таким чином, підтверджується можливість подання всіх моделей помилок (від 1 до 3) за допомогою залишку породжувального многочлена цього коду.

У табл. 11 реалізовано стандартну конфігурацію коду БЧХ (7, 1). Для кращого сприйняття стандартної конфігурації в табл. 11 виконано групування за моделями відстаней помилок.

Використовуючи метод декодування за моделлю відстані помилок, визначивши модель відстані, за допомогою якої виправляються помилки, неважко побудувати схему декодування.

Код БЧХ (7, 1) є кодом ТЕС, тому в табл. 10 наведено залишок і модель відстані, що виправляє помилки. За допомогою залишку, який відповідає цій моделі відстані, обчислюється конфігурація синдрому. Породжувальний многочлен для цього коду має вигляд

$$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1. \qquad (8)$$

Таблиця 11. Стандартна конфігурація кодів БЧХ (7, 1)

Модель відстані помилок	Кодоване слово	0000000	1111111
	Модель помилок	Кодова послідовність, що приймається	
—	0000000	0000000	1111111
(7)	0000001	0000001	1111110
	0000010	0000010	1111101
	0000100	0000100	1111011
	0001000	0001000	1110111
	0010000	0010000	1101111
	0100000	0100000	1011111
	1000000	1000000	0111111
(1, 6)	0000011	0000011	1111100
	0000110	0000110	1111001
	0001100	0001100	1110011
	0011000	0011000	1100111
	0110000	0110000	1001111
	1100000	1100000	0011111
	1000001	1000001	0111110
(2, 5)	0000101	0000101	1111010
	0001010	0001010	1110101
	0010100	0010100	1101011
	0101000	0101000	1010111
	1010000	1010000	0101111
	0100001	0100001	1011110
	1000010	1000010	0111101
(3, 4)	0001001	0001001	1110110
	0010010	0010010	1101101
	0100100	0100100	1011011
	1001000	1001000	0110111
	0010001	0010001	1101110
	0100010	0100010	1011101
	1000100	1000100	0111011
(1, 1, 5)	0000111	0000111	1111000
	0001110	0001110	1110001
	0011100	0011100	1100011
	0111000	0111000	1000111
	1110000	1110000	0001111
	1100001	1100001	0011110
	1000011	1000011	0111100
(1, 2, 4)	0001011	0001011	1110100
	0010110	0010110	1101001
	0101100	0101100	1010011
	1011000	1011000	0100111
	0110001	0110001	1001110
	1100010	1100010	0011101
	1000101	1000101	0111010
(1, 3, 3)	0010011	0010011	1101100
	0100110	0100110	1011001
	1001100	1001100	0110011
	0011001	0011001	1100110

Закінчення табл. 11

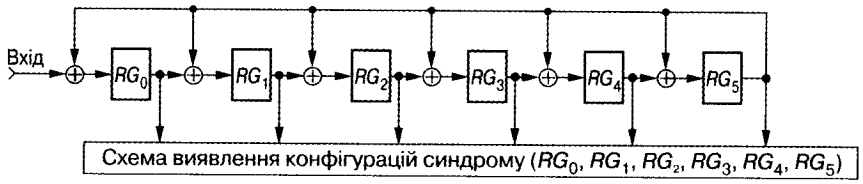
Модель відстані помилок	Кодоване слово	0000000	1111111
	Модель помилок	Кодова послідовність, що приймається	
(1, 3, 3)	0110010 1100100 1001001	0110010 1100100 1001001	1001101 0011011 0110110
(1, 4, 2)	0100011 1000110 0001101 0011010 0110100 1101000 1010001	0100011 1000110 0001101 0011010 0110100 1101000 1010001	1011100 0111001 1110010 1100101 1001011 0010111 0101110
(2, 2, 3)	0010101 0101010 1010100 0101001 1010010 0100101 1001010	0010101 0101010 1010100 0101001 1010010 0100101 1001010	1101010 1010101 0101011 1010110 0101101 1011010 0110101

Коренем многочлена $G(x)$ є α, тому $G(\alpha) = 0$. Тоді

$$\alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1. \qquad (9)$$

З формули (9) отримуємо, що

$$\alpha^6 = \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1. \qquad (10)$$



Модель відстані помилок	Множник	Конфігурація синдрому (RG_0, RG_1, RG_2, RG_3, RG_4, RG_5)
(7)	1	(1 0 0 0 0 0)
(1, 6)	$\alpha + 1$	(1 1 0 0 0 0)
(2, 5)	$\alpha^2 + 1$	(1 0 1 0 0 0)
(3, 4)	$\alpha^3 + 1$	(1 0 0 1 0 0)
(1, 1, 5)	$\alpha^2 + \alpha + 1$	(1 1 1 0 0 0)
(1, 2, 4)	$\alpha^3 + \alpha + 1$	(1 1 0 1 0 0)
(1, 3, 3)	$\alpha^4 + \alpha + 1$	(1 1 0 0 1 0)
(1, 4, 2)	$\alpha^5 + \alpha + 1$	(1 1 0 0 0 1)
(2, 2, 3)	$\alpha^4 + \alpha^2 + 1$	(1 0 1 0 1 0)

Рис. 25. Модель відстані помилок і конфігурація синдрому

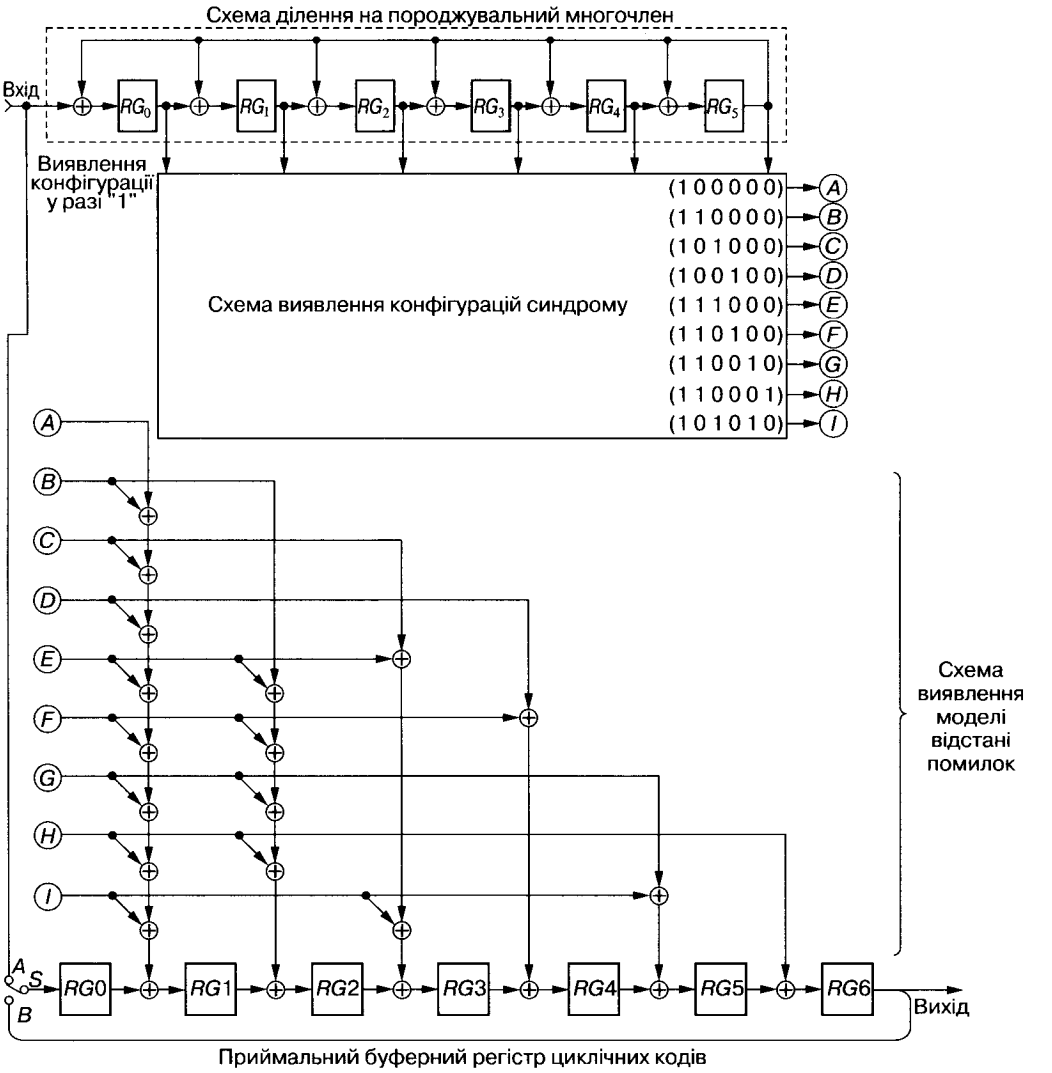


Рис. 26. Схема декодування за моделлю відстані помилок для кодів БЧХ (7, 1) ТЕС

Максимальний порядок залишку дорівнює 5. Отже, якщо в залишку моделі відстані помилок з'являється порядок вище шостого, то його можна знизити, використавши формулу (10). Звідси випливає, що у разі, коли знижено порядок породжувального многочлена, залишок дійсно можна обчислити за схемою ділення породжувального многочлена. Визначивши залишок за моделлю відстані помилок аж до потрійної, і використавши табл. 10, отримаємо, що максимальний порядок дорівнює 5. На рис. 25 наведено відповідність між конфігурацією синдрому і залишку для моделі відстані помилок.

Продемонструємо описаний вище принцип визначення конфігурації синдрому за моделлю відстані помилок на прикладі в більш доступній і зрозумілій формі.

Таблиця 12. Декодування за моделлю відстані

Перемикач	Такт	Схема ділення на породжувальний многочлен					
		RG_0	RG_1	RG_2	RG_3	RG_4	RG_5
B	0	$e_6 + e_0$	$e_6 + e_1$	$e_6 + (e_2)$	$e_6 + (e_3)$	$e_6 + e_4$	$e_6 + (e_5)$
	1	$e_6 + (e_5)$	$(e_5) + e_0$	$(e_5) + e_1$	$(e_5) + (e_2)$	$(e_5) + (e_3)$	$(e_5) + e_4$
	2	$(e_5) + e_4$	$e_6 + e_4$	$e_4 + e_0$	$e_4 + e_1$	$e_4 + (e_2)$	$e_4 + (e_3)$
	3	$e_4 + (e_3)$	$(e_5) + (e_3)$	$e_6 + (e_3)$	$(e_3) + e_0$	$(e_3) + e_1$	$(e_3) + (e_2)$
	4	$(e_3) + (e_2)$	$e_4 + (e_2)$	$(e_5) + (e_2)$	$e_6 + (e_2)$	$(e_2) + e_0$	$(e_2) + e_1$
	5	$(e_2) + e_1$	$(e_3) + e_1$	$e_4 + e_1$	$(e_5) + e_1$	$e_6 + e_1$	$e_1 + e_0$
	6	$e_1 + e_0$	$(e_2) + e_0$	$(e_3) + e_0$	$e_4 + e_0$	$(e_5) + e_0$	$e_6 + e_0$
	7	$e_6 + e_0$	$e_6 + e_1$	$e_6 + (e_2)$	$e_6 + (e_3)$	$e_6 + e_4$	$e_6 + (e_5)$
A	8	—	—	—	—	—	—
	9	—	—	—	—	—	—
	10	—	—	—	—	—	—
	11	—	—	—	—	—	—
	12	—	—	—	—	—	—
	13	—	—	—	—	—	—

Залишок, отриманий в регістрі схеми ділення породжувального многочлена, зростає від мінімального елемента α^0 зліва направо. Залишок, одержаний унаслідок обчислення за моделлю відстані помилок, є максимальним елементом зліва і зменшується направо. Наприклад, залишок у моделі відстані помилок (1, 4, 2) (див. табл. 10) описується виразом

$$R(\alpha) = \alpha^5 + \alpha + 1.$$

Для того, щоб значення порядку залишку в моделі відстані помилок зліва було нижчим, а справа — вищим, перепишемо многочлен залишку $R(\alpha)$ за зростаючими степенями α:

$$R(\alpha) = \alpha^5 + \alpha + 1 = \alpha^0 + \alpha^1 + \alpha^5.$$

Як наслідок отримаємо наступну відповідність конфігурації синдрому:

$$R(\alpha) = 1 \cdot \alpha^0 + 1 \cdot \alpha^1 + 0 \cdot \alpha^2 + 0 \cdot \alpha^3 + 0 \cdot \alpha^4 + 1 \cdot \alpha^5.$$

Неважко визначити конфігурацію синдрому, що виявляє цей залишок. У схему виявлення конфігурації синдрому необхідно ввести одиницю ("1") при виявленні конфігурації синдрому і ввести нуль ("0"), якщо такого виявлення не відбувається. Після виявлення і введення "1" формується модель відстані помилок, і помилка виправляється шляхом додавання за модулем 2 моделі відстані помилок і прийнятої кодової послідовності, що знаходиться в буферному регістрі.

Як приклад розглянемо роботу схеми декодування за моделлю відстані помилок для кодів БЧХ (7, 1) (рис. 26).

помилок кодів БЧХ (7, 1)

Вхід схеми виявлення конфігурації синдрому	Приймальний буферний регістр циклічних кодів							Вихід
	*RG*0	*RG*1	*RG*2	*RG*3	*RG*4	*RG*5	*RG*6	
(0 0 1 1 0 1)	$\alpha_0 + e_0$	$\alpha_0 + e_1$	$\alpha_0 + (e_2)$	$\alpha_0 + (e_3)$	$\alpha_0 + e_4$	$\alpha_0 + (e_5)$	$\alpha_0 + e_6$	—
(1 1 1 0 0 1)	$\alpha_0 + e_6$	$\alpha_0 + e_0$	$\alpha_0 + e_1$	$\alpha_0 + (e_2)$	$\alpha_0 + (e_3)$	$\alpha_0 + e_4$	$\alpha_0 + (e_5)$	—
(1 0 0 0 1 1)	$\alpha_0 + (e_5)$	$\alpha_0 + e_6$	$\alpha_0 + e_0$	$\alpha_0 + e_1$	$\alpha_0 + (e_2)$	$\alpha_0 + (e_3)$	$\alpha_0 + e_4$	—
(1 0 1 1 1 0)	$\alpha_0 + e_4$	$\alpha_0 + (e_5)$	$\alpha_0 + e_6$	$\alpha_0 + e_0$	$\alpha_0 + e_1$	$\alpha_0 + (e_2)$	$\alpha_0 + (e_3)$	—
(0 1 0 1 1 1)	$\alpha_0 + (e_3)$	$\alpha_0 + e_4$	$\alpha_0 + (e_5)$	$\alpha_0 + e_6$	$\alpha_0 + e_0$	$\alpha_0 + e_1$	$\alpha_0 + (e_2)$	—
(1 1 0 1 0 0)	$\alpha_0 + (e_2)$	$\alpha_0 + (e_3)$	$\alpha_0 + e_4$	$\alpha_0 + (e_5)$	$\alpha_0 + e_6$	$\alpha_0 + e_0$	$\alpha_0 + e_1$	—
Вихід виявленої конфігурації = 1	→ Вихід схеми виявлення моделі відстані помилок							
	1	1	(0)	1	(0)	(0)	(0)	
(0 1 1 0 1 0)	α_0	α_0	α_0	α_0	α_0	α_0	α_0	—
(0 0 1 1 0 1)	α_0	α_0	α_0	α_0	α_0	α_0	α_0	α_0
	—	α_0	α_0	α_0	α_0	α_0	α_0	α_0
	—	—	α_0	α_0	α_0	α_0	α_0	α_0
	—	—	—	α_0	α_0	α_0	α_0	α_0
	—	—	—	—	α_0	α_0	α_0	α_0
	—	—	—	—	—	α_0	α_0	α_0
	—	—	—	—	—	—	α_0	α_0

$((e_5), (e_3), (e_2) = 1; e_6, e_4, e_1, e_0 = 0)$

У табл. 12 наведено дії, які виконуються при появі потрійної помилки в другому, четвертому і п'ятому бітах. Модель таких помилок має вигляд (0101100), а модель відстані помилок — (1, 2, 4). Конфігурація синдрому для виявлення цієї моделі відстані помилок задається виразом

$$(RG_0, RG_1, RG_2, RG_3, RG_4, RG_5) = (1\ 1\ 0\ 0\ 0\ 1).$$

Таким чином, спосіб декодування за моделлю відстані помилок є способом декодування циклічних кодів з виправленням багатократних помилок. Коди БЧХ є характерними представниками циклічних кодів, що виправляють помилки, тому для них можна використовувати описаний спосіб декодування.

18.1. ДЕКОДУВАННЯ КОДІВ БЧХ, ВИКОРИСТОВУВАНИХ НА ПРАКТИЦІ

Описаний вище принцип декодування за моделлю відстані помилок розроблений з урахуванням методів декодування лінійних кодів і властивостей циклічних кодів. Він є загальноприйнятим принципом декодування циклічних кодів.

Повторимо, що у разі декодування за моделлю відстані помилок виправлення помилок здійснюється за допомогою такої процедури:

1. Визначається залишок породжувального многочлена.

2. За допомогою циркуляції залишку в ділильному ланцюзі породжувального многочлена визначається конфігурація синдрому для моделі відстані помилок.

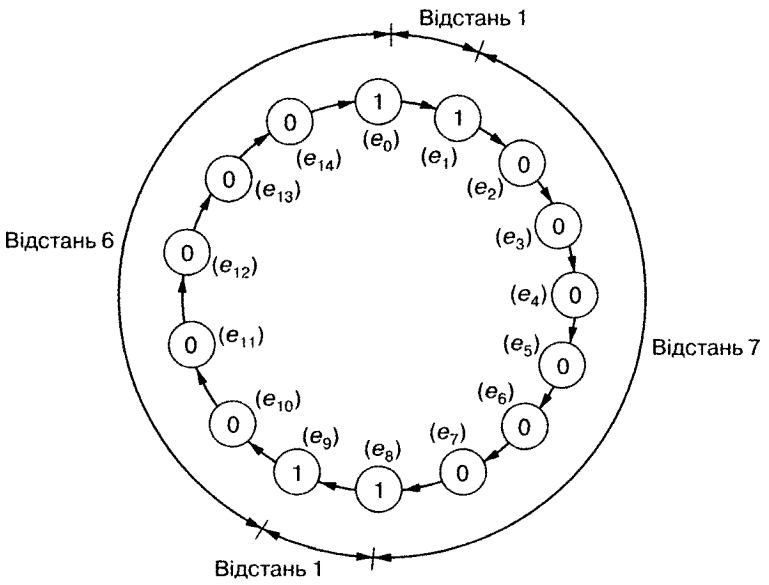


Рис. 27. Модель відстаней між помилковими розрядами

3. Модель (структура) відстані помилок, яка відповідає виявленій конфігурації (структурі) синдрому, за допомогою підсумовування за модулем 2 додається до кодової послідовності, що приймається і циркулює синхронно із залишком у циклічному приймальному буферному регістрі.

Таким чином, важливим завданням у разі декодування за моделлю відстані помилок є дослідження моделі відстаней помилок. Розглянемо модель відстаней помилок для коду БЧХ з числом кодових розрядів n. Модель помилок сформована з числа розрядів n, яке збігається з числом кодових розрядів. У цій n-розрядній моделі помилок передбачається, що l ($l \le n$) розрядів є помилковими.

У даному випадку модель відстаней помилок знаходиться таким чином. Циклічно формується модель помилок: за годинниковою стрілкою вимірюються відстані між помилковими бітами, виконується циклічна перестановка цих відстаней, які потім розташовуються в такому порядку: відстань d_1, відстань d_2, ..., відстань d_l; де $d_1 \le d_2, d_3, ..., d_l$. Цей ряд відстаней вважається моделлю (структурою) відстаней ($d_1, d_2, ..., d_l$).

Вважаючи, що здатність до виправлення помилок для такого коду БЧХ полягає у виправленні t-кратних помилок, маємо для дійсно знайденої моделі відстаней, що $l \le t$. Іншими словами, отримуємо модель відстаней аж до t-кратних помилок.

Наприклад, спробуємо визначити модель відстаней для чотирикратної помилки (помилкові 6, 7, 14 і 15-й біти) у разі коду БЧХ з 15 кодовими розрядами. Модель помилок має вигляд (e_9, e_8, e_1, e_0).

При циклічному поданні моделі помилок модель відстаней між помилковими розрядами наведена на рис. 27.

Згідно з рис. 27, відстані між помилковими бітами є такими:

При цьому є чотири послідовності відстаней:
1) відстань 1, відстань 7, відстань 1, відстань 6;
2) відстань 7, відстань 1, відстань 6, відстань 1;
3) відстань 1, відстань 6, відстань 1, відстань 7;
4) відстань 6, відстань 1, відстань 7, відстань 1.

Серед цих чотирьох послідовностей у порядку зростання відстані між помилковими бітами побудовано послідовність 3, тоді модель відстаней помилок можна записати так: (1, 6, 1, 7).

Наведений приклад пояснює процес пошуку моделі відстаней помилок. Побудуємо схеми декодування за принципом визначення моделі відстаней помилок для кодів БЧХ, які часто використовуються на практиці:

- SEC-DED;
- DEC (15,7);
- TEC (15,5);
- DEC (31,21).

18.1.1. Коди БЧХ SEC-DED

Виконуючи передачу супутником цифрових звукових сигналів, використовують такі два коди БЧХ: (7, 3) SEC-DED; (63,56) SEC-DED. Ці коди БЧХ дозволяють виявляти подвійні помилки. Вартою уваги є модель відстані помилок у разі їх виявлення.

Спочатку розглянемо код БЧХ (7, 3) SEC-DED. Цей код вже розглядався раніше як код Хеммінга, що виправляє помилки (ч. 1, розд. 14). Проте, доки не відбулося кодування початково як коду БЧХ, бажано ще раз розглянути цей код в іншій якості.

Між числом розрядів коду n і порядком початкового многочлена M існує співвідношення

$$n = 2^M - 1.$$

Тому у разі, коли число кодових розрядів дорівнює 7, порядок початкового многочлена — 3.

Як многочлен третього порядку вибрано многочлен $x^3 + x + 1$, причому його коренем є α. Оскільки здатність до виправлення помилок відповідає SEC (виправлення одиничних помилок), послідовне піднесення параметра α до степеня дає: α, α^2.

При додаванні α^0 для виявлення помилок виконується послідовне піднесення до степеня α для SEC-DED: α^0, α, α^2. Породжувальний многочлен міс-

тить ці послідовні степені як корені, тому, якщо знайти для нього мінімальні многочлени, то:

- для α^0 мінімальним многочленом є $x + 1$;
- для α мінімальним многочленом є $x^3 + x + 1$;
- для α^2 мінімальним многочленом є $x^3 + x + 1$.

Отже, породжувальний многочлен має вигляд

$G(x) = \{$*породжувальний многочлен для* α^0, *породжувальний многочлен для* α, *породжувальний многочлен для* $\alpha^2\} = \{x + 1, x^3 + x + 1, x^3 + x + 1\} = (x + 1)(x^3 + x + 1) = x^4 + x^3 + x^2 + 1,$

тобто він є простим добутком мінімальних многочленів.

Порядок породжувального многочлена дорівнює 4, тому число контрольних розрядів (бітів) становить 4, а число інформаційних розрядів — 3.

Код, що отримується на базі породжувального многочлена, коренями якого є послідовні степені α^0, α, α^2 кореня α початкового многочлена $x^3 + x + 1$, — це код БЧХ (7, 3), у якого:

- число розрядів $n = 7$;
- число контрольних розрядів $m = 4$;
- число інформаційних розрядів $k = 3$;
- є здатність до виправлення помилок SEC-DED;
- породжувальний многочлен — $x^4 + x^3 + x^2 + 1$.

Інформаційні розряди і контрольні розряди формуються таким чином:

Інформаційні розряди			Контрольні розряди			
a_2	a_1	a_0	c_3	c_2	c_1	c_0
1	2	3	4	5	6	7

При цьому контрольні розряди пов'язані з інформаційними розрядами такими співвідношеннями:

$$c_3 = a_2 + a_0 \ (\text{mod}2),$$
$$c_2 = a_2 + a_1 + a_0 \ (\text{mod}2),$$
$$c_1 = a_2 + a_1 \ (\text{mod}2),$$
$$c_0 = a_1 + a_0 \ (\text{mod}2).$$

На рис. 28 показано, в які з контрольних розрядів включені інформаційні розряди. Якщо розглянути число розрядів, що змінюються в усьому кодованому слові у разі зміни інформаційних розрядів, то можна визначити кодову відстань. Як бачимо з рисунка, мінімальна кодова відстань дорівнює 4. Тому, виходячи з поданих на рис. 28 кодових відстаней, зрозуміло, що здатність такого коду до виправлення помилок дійсно відповідає SEC-DED.

Для формування синдрому і встановлення його відповідності положенню помилок виконуємо наступні процедури.

Спочатку декодуємо код БЧХ (7, 3) як код Хеммінга. Тоді для цього коду є два контрольні стовпці: $x + 1$ та $x^3 + x + 1$.

Якщо контрольним стовпцем є $x + 1$, то використовується послідовність α^0, а саме:

$$(\alpha^0)^0, (\alpha^0)^1, (\alpha^0)^2, (\alpha^0)^3, (\alpha^0)^4, (\alpha^0)^5, (\alpha^0)^6.$$

Інформаційні розряди	Контрольні розряди, що містять інформаційні розряди	Загальне число розрядів, що змінюються при зміні інформаційного розряду
a_2	c_3, c_2, c_1	4
a_1	c_2, c_1, c_0	4
a_0	c_3, c_2, c_0	4

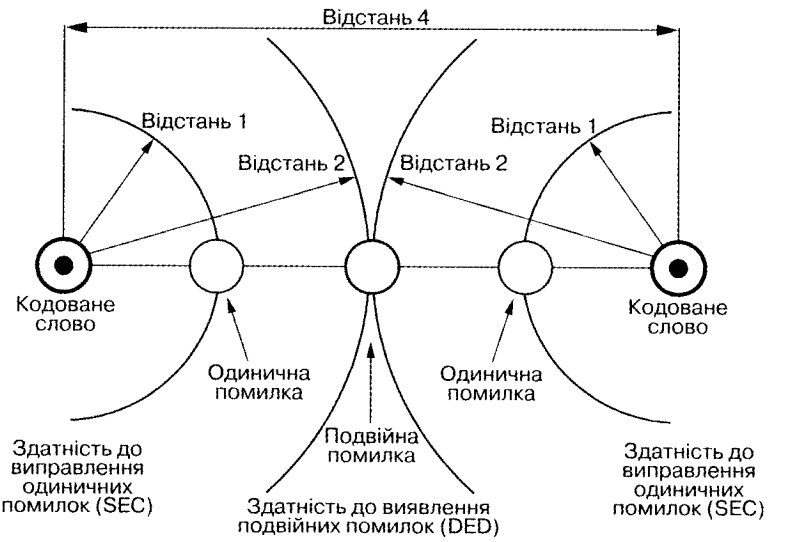


Рис. 28. Виявлення і виправлення помилок при кодовій відстані 4 для коду БЧХ (7, 3)

Якщо контрольним стовпцем є $x^3 + x + 1$, то використовується послідовність α, а саме:

$$(\alpha)^0, (\alpha)^1, (\alpha)^2, (\alpha)^3, (\alpha)^4, (\alpha)^5, (\alpha)^6.$$

За допомогою цих двох контрольних стовпців, можна сформувати синдром. Відповідність між синдромами і положенням помилок наведена в табл. 13.

Згідно з табл. 13, матриця H перевірки на парність коду БЧХ (7, 3) набуває такого вигляду:

$$H = \begin{pmatrix} (\alpha^0)^6, & (\alpha^0)^5, & (\alpha^0)^4, & (\alpha^0)^3, & (\alpha^0)^2, & (\alpha^0)^1, & (\alpha^0)^0 \\ \alpha^6, & \alpha^5, & \alpha^4, & \alpha^3, & \alpha^2, & \alpha^1, & \alpha^0 \end{pmatrix} =$$

$$= \begin{pmatrix} 1, & 1, & 1, & 1, & 1, & 1, & 1 \\ \alpha^6, & \alpha^5, & \alpha^4, & \alpha^3, & \alpha^2, & \alpha^1, & \alpha^0 \end{pmatrix}. \qquad (11)$$

Якщо помилкову послідовність записати так: $e = (e_6, e_5, e_4, e_3, e_2, e_1, e_0)$, то синдром S можна відшукати у вигляді

$$S = He^T.$$

Оскільки матриця перевірки на парність H складається з двох рядків і якщо S_0, S_1 — розряди синдромів, які відповідають рядкам матриці, то синдром на-

буває вигляду

$$S = \begin{pmatrix} S_1 \\ S_0 \end{pmatrix} = He^T = \begin{pmatrix} 1, & 1, & 1, & 1, & 1, & 1, & 1 \\ \alpha^6, & \alpha^5, & \alpha^4, & \alpha^3, & \alpha^2, & \alpha, & 1 \end{pmatrix} e^T. \qquad (12)$$

Виходячи з цього, для розрядів синдромів S_0, S_1 отримуємо відповідно:

$$S_0 = \left(\alpha^6, \alpha^5, \alpha^4, \alpha^3, \alpha^2, \alpha^1, 1\right) e^T = \left(\alpha^6, \alpha^5, \alpha^4, \alpha^3, \alpha^2, \alpha^1, 1\right) \begin{pmatrix} e_6 \\ e_5 \\ e_4 \\ e_3 \\ e_2 \\ e_1 \\ e_0 \end{pmatrix} =$$

$$= e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0, \qquad (13)$$

$$S_1 = (1,1,1,1,1,1,1) e^T = (1,1,1,1,1,1,1) \begin{pmatrix} e_6 \\ e_5 \\ e_4 \\ e_3 \\ e_2 \\ e_1 \\ e_0 \end{pmatrix} =$$

$$= e_6 + e_5 + e_4 + e_3 + e_2 + e_1 + e_0. \qquad (14)$$

З виразу (13) для розрядів синдрому S_0 зрозуміло, що він є синдромом, призначеним для виправлення одиничних помилок (див. ч. 1). Крім того, із співвідношення (14) для розрядів синдрому S_0 бачимо, що він є синдромом загальної перевірки на парність. Іншими словами, в цьому випадку відповідні розряди синдромів S_0, S_1 мають різне призначення і тому іменуються таким чином:

S_0 — SEC (виявлення положення помилкового розряду);

S_1 — загальна перевірка парності (виявлення парного числа, непарного числа для помилкових розрядів).

Отже, в цьому випадку схема декодування може бути побудована за допомогою об'єднання схем виявлення помилки, створених для кожної функції розрядів синдромів (мінімальних многочленів $x + 1$ та $x^3 + x + 1$).

На рис. 29 наведено схему декодування коду БЧХ (7, 3) SEC-DED. У ній

Таблиця 13. Синдроми і положення помилок

Синдром		Положення помилок
S_1	S_0	
0	0	Помилкові розряди відсутні
$(\alpha^0)^6 = 1$	α^6	Перший розряд
$(\alpha^0)^5 = 1$	α^5	Другий розряд
$(\alpha^0)^4 = 1$	α^4	Третій розряд
$(\alpha^0)^3 = 1$	α^3	Четвертий розряд
$(\alpha^0)^2 = 1$	α^2	П'ятий розряд
$(\alpha^0)^1 = 1$	α^1	Шостий розряд
$(\alpha^0)^0 = 1$	$\alpha^0 = 1$	Сьомий розряд

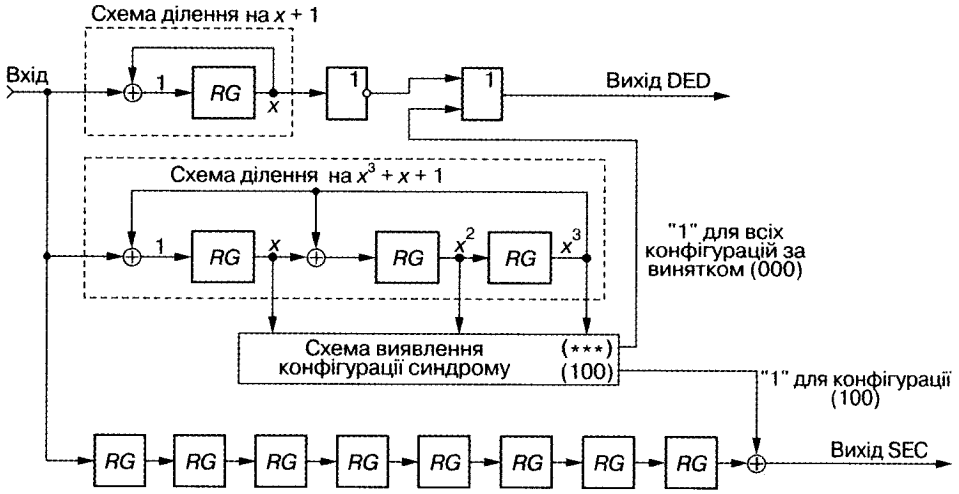


Рис. 29. Схема декодування коду БЧХ (7, 3)

використовуються:

- схема ділення $x + 1$, яка є схемою перевірки парності, або схемою визначення розрядів синдрому S_1;
- схема ділення $x^3 + x + 1$, яка розраховує залишок для SEC, іншими словами, є схемою визначення розрядів синдрому S_0;
- схема визначення конфігурації синдрому, призначена для виявлення положення помилок.

Далі спробуємо виконати декодування цього коду з використанням принципу декодування за моделлю відстані помилок. Число кодових розрядів коду БЧХ (7, 3) дорівнює 7, причому число контрольних розрядів — 4, тому число станів контролю, які дозволяють відобразити ці контрольні розряди, — $2^4 = 16$.

Цей код є кодом з виправленням одиничних помилок (SEC), тому можуть бути виправлені всі одиничні помилки. Отже, коли повністю сформовано стандартну конфігурацію, як характерна база може бути вибрана конфігурація з одиничною помилкою

Кількість варіантів конфігурації з одиничною помилкою в цілому становить $\left(\frac{7}{1}\right) = \frac{7!}{1!6!} = 7$. Ця конфігурація з одиничною помилкою є моделлю відстаней помилок (7).

Число конфігурацій без помилок становить $\left(\frac{7}{0}\right) = \frac{7!}{0!7!} = 1$. Отже, число помилкових конфігурацій, які можуть бути виправлені, дорівнює сумі конфігурацій без помилок і конфігурацій з одиничною помилкою: $\left(\frac{7}{0}\right) + \left(\frac{7}{1}\right) = 1 + 7 = 8$.

Число, яке можна записати контрольними розрядами, дорівнює 16, тому, якщо подати стани без помилок і з одиничною помилкою, то одержимо залишок: 16 – 8 = 8. Іншими словами, за допомогою залишку можна подати

тільки 8 станів контролю. Тому виникає проблема щодо можливості виявлення подвійних помилок.

Неважко показати, що подвійні помилки можна повністю виявити, якщо для моделі відстаней (j, k) у разі подвійних помилок вибрати співвідношення $j + k = 7$, $j \le k$. Якщо розглянути всі конфігурації (j, k) як моделі відстаней для подвійних помилок, то наявними є три з них:

- модель відстаней помилок (1, 6);
- модель відстаней помилок (2, 5);
- модель відстаней помилок (3, 4).

За допомогою однієї такої моделі відстаней помилок можна виявити сім конфігурацій подвійних помилок. У табл. 14 наведено моделі відстаней для подвійних помилок, а також конфігурації подвійних помилок, які можуть бути виявлені за допомогою цих моделей.

Таким чином, є три моделі відстаней помилок, за допомогою яких можна виявити помилкову конфігурацію. Загальне число конфігурацій з подвійними помилками становить

$$\binom{7}{2} = \frac{7!}{2!(7-2)!} = \frac{6 \cdot 7}{2} = 21.$$

Отже, за допомогою трьох моделей відстаней помилок можуть бути виявлені всі конфігурації з подвійними помилками.

Для станів контролю залишилося тільки вісім комбінацій. Тому даний код є прийнятним, якщо потрібно не виправляти подвійні помилки, а лише їх виявляти. За допомогою восьми станів контролю можна виявити тільки одну модель відстаней подвійних помилок. Проте можна припустити наявність можливості виявлення двох інших моделей відстаней помилок з використанням однієї і тієї самої моделі. На підставі циркуляції помилкових конфігурацій, що належать відповідним моделям відстаней помилок, спробуємо оцінити їх взаємозв'язок.

Для коду, у якого як початковий многочлен використовується $x^3 + x + 1$ з коренем α, залишок можна подати так:

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0.$$

Крім того, α є примітивним коренем, який формує кінцеве поле $GF(2^3)$, залишку початкового многочлена $x^3 + x + 1$, отже, $\alpha^7 = 1$.

Крім цього, α є одним з коренів породжувального многочлена $x^4 + x^3 + x^2 + 1$, тому $\alpha^4 + \alpha^3 + \alpha^2 + 1 = 0$. Отже, маємо співвідношення

$$\alpha^4 = \alpha^3 + \alpha^2 + 1. \qquad (15)$$

Спочатку проаналізуємо конфігурацію помилок для моделі відстаней помилок (1, 6). Залишок такої моделі (1, 6) має вигляд

$$R(\alpha) = \alpha + 1.$$

При цьому конфігурацією помилок є (e_1, e_0).

У разі множення залишку на корінь α, тобто при циркуляції залишку, має місце наступна відповідність між конфігураціями і моделями відстаней помилок:

Залишок	Конфігурація помилок	Модель відстаней помилок
$R(\alpha) = \alpha + 1$ $\alpha R(\alpha) = \alpha^2 + \alpha$ $\alpha^2 R(\alpha) = \alpha^3 + \alpha^2$ $\alpha^3 R(\alpha) = \alpha^4 + \alpha^3 =$ $= (\alpha^3 + \alpha^2 + 1) + \alpha^3 =$ $= \alpha^2 + 1$	(e_1, e_0) (e_2, e_1) (e_3, e_2) (e_4, e_3) (e_2, e_0)	(1, 6)
$\alpha^4 R(\alpha) = \alpha^5 + \alpha^4 =$ $= \alpha(\alpha^3 + \alpha^2 + 1)\alpha^4 =$ $= \alpha^3 + \alpha$	(e_5, e_4) (e_3, e_1)	(2, 5)
$\alpha^5 R(\alpha) = \alpha^6 + \alpha^5 =$ $= \alpha^2(\alpha^3 + \alpha^2 + 1) + \alpha^5 =$ $= \alpha^4 + \alpha^2 =$ $= (\alpha^3 + \alpha^2 + 1) + \alpha^2 =$ $= \alpha^3 + 1$	(e_6, e_5) (e_3, e_0)	(2, 5)
$\alpha^6 R(\alpha) = \alpha^7 + \alpha^6 =$ $= \alpha^6 + 1 =$ $= \alpha^2(\alpha^3 + \alpha^2 + 1) + 1 =$ $= \alpha^5 + \alpha^4 + \alpha^2 + 1 =$ $= \alpha(\alpha^3 + \alpha^2 + 1) + \alpha^4 + \alpha^2 + 1 =$ $= \alpha^4 + \alpha^3 + \alpha + \alpha^4 +$ $+ \alpha^2 + 1 =$ $= \alpha^3 + \alpha^2 + \alpha + 1 =$ $= \alpha^4 + \alpha$	(e_6, e_0) (e_4, e_1)	(3, 4)
$\alpha^7 R(\alpha) = \alpha^8 + \alpha^7 =$ $= \alpha + 1$	(e_1, e_0)	(3, 4)

Якщо спробувати здійснити циркуляцію помилкової конфігурації для моделі відстаней помилок (1, 6), то в межах одного циклу виникають залишки помилкових конфігурацій, що належать іншим моделям, а саме:

залишок $(e_4, e_3) = \alpha^4 + \alpha^3 = \alpha^2 + 1$ = залишок (e_2, e_0) ← модель відстаней помилок (2, 5);

залишок = $(e_6, e_5) = \alpha^6 + \alpha^5 = \alpha^3 + 1$ = залишок (e_3, e_0) ← модель відстаней помилок (3, 4).

Це вказує на те, що неможливо розрізнити три моделі відстаней подвійних помилок.

У табл. 15 наведені залишки конфігурацій подвійних помилок, а також відповідні цим залишкам помилкові конфігурації. Виходячи з цієї таблиці зрозуміло, що одному залишку відповідають три конфігурації подвійних помилок.

Таблиця 14. Моделі відстаней і конфігурації помилок

Модель відстаней помилок	Конфігурації помилок, які можуть бути виявлені
(1, 6)	(e_6, e_0), (e_6, e_5), (e_5, e_4), (e_4, e_3), (e_3, e_2), (e_2, e_1), (e_1, e_0)
(2, 5)	(e_6, e_1), (e_5, e_0), (e_6, e_4), (e_5, e_3), (e_4, e_2), (e_3, e_1), (e_2, e_0)
(3, 4)	(e_6, e_2), (e_5, e_1), (e_4, e_0), (e_6, e_3), (e_5, e_2), (e_4, e_1), (e_3, e_0)

Таблиця 15. Залишки подвійних помилок і помилкові конфігурації

Залишок подвійної помилки	Помилкова конфігурація моделі відстаней помилок		
	(1, 6)	(2, 5)	(3, 4)
$\alpha + 1$	(e_1, e_0)	(e_6, e_4)	(e_5, e_2)
$\alpha^2 + \alpha$	(e_2, e_1)	(e_5, e_0)	(e_6, e_3)
$\alpha^3 + \alpha$	(e_3, e_4)	(e_6, e_1)	(e_4, e_0)
$\alpha^2 + \alpha$	(e_4, e_3)	(e_2, e_0)	(e_5, e_1)
$\alpha^3 + \alpha$	(e_5, e_4)	(e_3, e_1)	(e_6, e_2)
$\alpha^3 + 1$	(e_6, e_5)	(e_4, e_2)	(e_3, e_0)
$\alpha^3 + \alpha^2 + \alpha + 1$	(e_6, e_0)	(e_5, e_3)	(e_4, e_1)

Накладення трьох моделей відстаней подвійних помилок вказує, що у разі їх виникнення виявлення можливе за допомогою лише однієї моделі. З трьох моделей відстаней помилок для виявлення подвійних помилок скористаємося моделлю (1, 6).

Утворюючи стандартну конфігурацію коду БЧХ (7, 3) на підставі розглянутих вище міркувань, як характерний базис можна використовувати наступні 15 конфігурацій:

- одна конфігурація без помилок;
- сім конфігурацій помилок моделі конфігурацій помилок (7);
- сім конфігурацій помилок моделі конфігурацій помилок (1, 6).

Оскільки уже вибрано конфігурації одиничних і подвійних помилок, за допомогою одного базису (представницького елемента), що лишився, можна відобразити конфігурації потрійних помилок. Водночас спроба вибору конфігурацій потрійних помилок як тільки одного базису позбавлена сенсу. У табл. 16 відображено створення стандартних конфігурацій коду БЧХ (7, 3) з використанням як базисів конфігурацій одиничних помилок з можливістю їх виправлення та конфігурацій подвійних помилок з можливістю їх виявлення.

Таким чином, конфігурації одиничних помилок можуть бути повністю виявлені за допомогою залишку $R(\alpha)$ моделі відстаней помилок (7), а конфігурації подвійних помилок — за допомогою залишку $R(\alpha)$ моделі відстаней помилок (1, 6). Порядок цих залишків не перевищує 3. Отже, виходячи з цих залишків, конфігурація синдрому моделі відстаней помилок (7) записується в такому вигляді:

$$(S\alpha_0, S\alpha_1, S\alpha_2, S\alpha_3) = (1\ 0\ 0\ 0),$$

а конфігурація синдрому моделі відстаней помилок (1, 6) —

$$(S\alpha_0, S\alpha_1, S\alpha_2, S\alpha_3) = (1\ 1\ 0\ 0).$$

Після визначення конфігурації синдромів для виявлення моделей відстаней помилок схема декодування коду БЧХ (7, 3) набуває вигляду, поданого на рис. 30.

У табл. 17 та 18 наведено відповідно операції щодо виправлення одиничних помилок (SEC) і виявлення подвійних помилок (DED) з моменту завершення введення вхідної кодової послідовності, тобто з моменту відшукання залишку

$$R(\alpha) = e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0.$$

Залишок, отриманий у момент завершення введення вхідної кодової послідовності, як вихідного сигналу схеми ділення на породжувальний многочлен,

Таблиця 16. Стандартні конфігурації коду БЧХ (7, 3)

Модель відстаней помилок	Конфігурація помилок	Інформаційне слово							
		000	001	010	100	011	101	110	111
		Кодове слово, що передається							
		0000000	0011101	0100111	1001110	0111010	1010011	1101001	1110100
		Кодове слово, що приймається							
—	0000000	0000000	0011101	0100111	1001110	0111010	1010011	1101001	1110100
(7)	0000001	0000001	0011100	0100110	1001111	0111011	1010010	1101000	1110101
	0000010	0000010	0011111	0100101	1001100	0111000	1010001	1101011	1110110
	0000100	0000100	0011001	0100011	1001010	0111110	1010111	1101101	1110000
	0001000	0001000	0010101	0101111	1000110	0110010	1011011	1100001	1111100
	0010000	0010000	0001101	0110111	1011110	0101010	1000011	1111001	1100100
	0100000	0100000	0111101	0000111	1101110	0011010	1110011	1001001	1010100
	1000000	1000000	1011101	1100111	0001110	1111010	0010011	0101001	0110100
(1, 6)	0000011	0000011	0011110	0100100	1001101	0111001	1010000	1101010	1110111
	0000110	0000110	0011011	0100001	1001000	0111100	1010101	1101111	1110010
	0001100	0001100	0010001	0101011	1000010	0110100	1011111	1100101	1111000
	0011000	0011000	0000101	0111111	1010110	0100010	1001011	1110001	1101100
	0110000	0110000	0101101	0010111	1111110	0001010	1100011	1011001	1000100
	1100000	1100000	1111101	1000111	0101110	1011010	0110011	0001001	0010100
	1000001	1000001	1011100	1100110	0001111	1111011	0010010	0101000	0110101

відображається многочленом порядку не вище ніж 3. Використовуючи співвідношення (15), маємо

$$R(\alpha) = (e_6 + e_4 + e_3)\alpha^3 + (e_6 + e_5 + e_4 + e_2)\alpha^2 + (e_6 + e_5 + e_1)\alpha + (e_5 + e_4 + e_0).$$

Код БЧХ (7, 3) має здатність до виправлення одиничних помилок (SEC), тому, як зазначалося вище, особливої необхідності у використанні циклічного вхідного буферного регістра немає. Отже, тут можуть застосовуватися звичайні приймальні буферні регістри. Така схема декодування наведена на рис. 31.

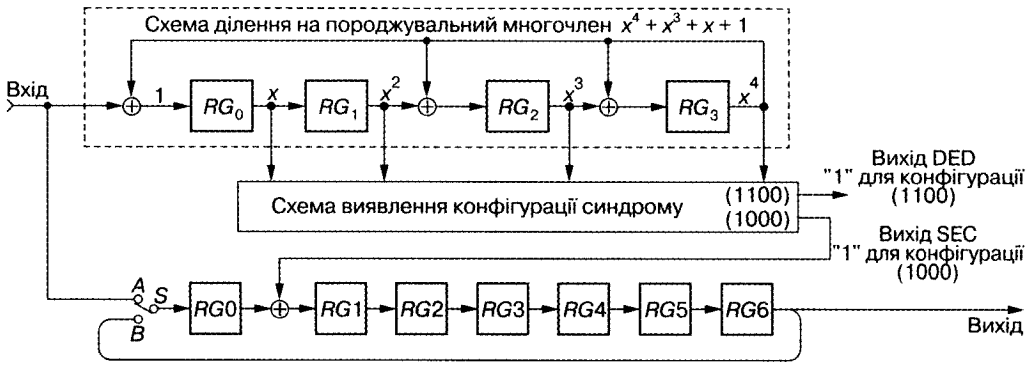


Рис. 30. Схема декодування коду БЧХ (7, 3) з циклічним буферним регістром

Таблиця 17. Операції з виправлення одиничних помилок (SEC)

Перемикач	Такт	Схема ділення на породжувальний многочлен				Вихід SEC	Циклічний вхідний буферний регістр							Вихід
		RG_0	RG_1	RG_2	RG_3		$RG0$	$RG1$	$RG2$	$RG3$	$RG4$	$RG5$	$RG6$	
	0	$e_5 + e_4 + (e_0)$	$e_6 + e_5 + e_1$	$e_6 + e_5 + e_4 + e_2$	$e_6 + e_4 + e_3$	e_0	$c_0 + e_0$	$c_1 + e_1$	$c_2 + e_2$	$c_3 + e_3$	$a_0 + e_4$	$a_1 + e_5$	$a_2 + e_6$	—
	1	$(e_6) + e_4 + e_3$	$e_5 + e_4 + e_0$	$e_5 + e_4 + e_3 + e_1$	$e_5 + e_3 + e_2$	e_6	$a_2 + e_6$	c_0	$c_1 + e_1$	$c_2 + e_2$	$c_3 + e_3$	$a_0 + e_4$	$a_1 + e_5$	—
	2	$(e_5) + e_3 + e_2$	$e_6 + e_4 + e_3$	$e_4 + e_3 + e_2 + e_0$	$e_4 + e_2 + e_1$	e_5	$a_1 + e_5$	a_2	c_0	$c_1 + e_1$	$c_2 + e_2$	$c_3 + e_3$	$a_0 + e_4$	—
B	3	$(e_4) + e_2 + e_1$	$e_5 + e_3 + e_2$	$e_6 + e_3 + e_2 + e_1$	$e_3 + e_1 + e_0$	e_4	$a_0 + e_4$	a_1	a_2	c_0	$c_1 + e_1$	$c_2 + e_2$	$c_3 + e_3$	—
	4	$(e_3) + e_1 + e_0$	$e_4 + e_2 + e_1$	$e_5 + e_2 + e_1 + e_0$	$e_6 + e_2 + e_0$	e_3	$c_3 + e_3$	a_0	a_1	a_2	c_0	$c_1 + e_1$	$c_2 + e_2$	—
	5	$e_6 + (e_2) + e_0$	$e_3 + e_1 + e_0$	$e_6 + e_4 + e_1 + e_0$	$e_6 + e_5 + e_1$	e_2	$c_2 + e_2$	c_3	a_0	a_1	a_2	c_0	$c_1 + e_1$	—
	6	$e_6 + e_5 + (e_1)$	$e_6 + e_2 + e_0$	$e_6 + e_5 + e_3 + e_0$	$e_5 + e_4 + e_0$	e_1	$c_1 + e_1$	c_2	c_3	a_0	a_1	a_2	c_0	—
	7	—	—	—	—	—	c_0	c_1	c_2	c_3	a_0	a_1	a_2	a_2
	8	—	—	—	—	—	—	c_0	c_1	c_2	c_3	a_0	a_1	a_1
	9	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	a_0	a_0
A	10	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_3
	11	—	—	—	—	—	—	—	—		c_0	c_1	c_2	c_2
	12	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_1
	13	—	—	—	—	—	—	—	—	—	—	—	c_0	c_0

У послідовності e_6—e_0 тільки один біт дорівнює "1"

Таблиця 18. Операції з виявлення подвійних помилок (DED)

Такт	Схема ділення на породжувальний многочлен				Конфігурації подвійних помилок, що виявляються конфігурацією синдрому (1100)		
	RG_0	RG_1	RG_2	RG_3			
0	$e_5 + e_4 + e_0$	$e_6 + e_5 + e_1$	$e_6 + e_5 + e_4 + e_2$	$e_6 + e_4 + e_3$	(e_1, e_0)	(e_6, e_4)	(e_5, e_2)
1	$e_6 + e_4 + e_3$	$e_5 + e_4 + e_0$	$e_5 + e_4 + e_3 + e_1$	$e_5 + e_3 + e_2$	(e_6, e_0)	(e_5, e_3)	(e_4, e_1)
2	$e_5 + e_3 + e_2$	$e_6 + e_4 + e_3$	$e_4 + e_3 + e_2 + e_0$	$e_4 + e_2 + e_1$	(e_6, e_5)	(e_4, e_2)	(e_3, e_0)
3	$e_4 + e_2 + e_1$	$e_5 + e_3 + e_2$	$e_6 + e_3 + e_2 + e_1$	$e_3 + e_1 + e_0$	(e_5, e_4)	(e_3, e_1)	(e_6, e_2)
4	$e_3 + e_1 + e_0$	$e_4 + e_2 + e_1$	$e_5 + e_2 + e_1 + e_0$	$e_6 + e_2 + e_0$	(e_4, e_3)	(e_2, e_0)	(e_5, e_1)
5	$e_6 + e_2 + e_0$	$e_3 + e_1 + e_0$	$e_6 + e_4 + e_1 + e_0$	$e_6 + e_5 + e_1$	(e_3, e_2)	(e_6, e_1)	(e_4, e_0)
6	$e_6 + e_5 + e_1$	$e_6 + e_2 + e_0$	$e_6 + e_5 + e_3 + e_0$	$e_5 + e_4 + e_0$	(e_2, e_1)	(e_5, e_0)	(e_6, e_3)

У послідовності e_6—e_0 не більше ніж два біти дорівнюють "1"

Для коду БЧХ (63, 56) SEC-DED, який використовується у разі супутникової передачі цифрових сигналів, схема декодування є аналогічною.

Код БЧХ (63, 56) SEC-DED — це код, отриманий з породжувального многочлена $x^7 + x^6 + x^2 + 1$, корені якого — α^0, α^1, α^2 — зростаючі степені кореня α початкового многочлена $x^6 + x + 1$.

Число контрольних розрядів цього коду дорівнює 7, отже, можна відобразити $2^7 = 128$ контрольних станів. Число кодових розрядів дорівнює 63, тому маємо таку загальну кількість конфігурацій одиничних помилок:

$$\left(\frac{63}{1}\right) = \frac{63!}{1!(63-1)!} = 63.$$

Модель відстаней помилок для конфігурацій одиничних помилок має вигляд (63). За її допомогою можна виявити всі одиничні помилки.

Наведений код здатний виправляти одиничні помилки (SEC), тому характерним базисом можна вибрати конфігурацію без помилок і всі конфігурації з одиничними помилками. Загальне число таких конфігурацій становить

$$\left(\frac{63}{0}\right) + \left(\frac{63}{1}\right) = 1 + 63 = 64.$$

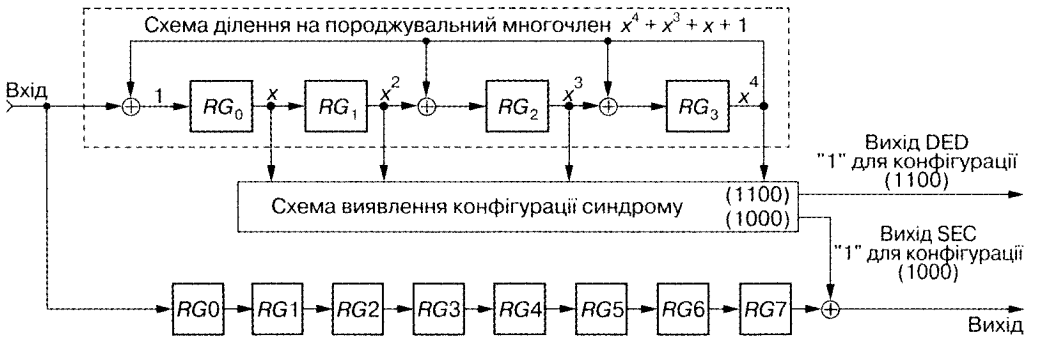


Рис. 31. Схема декодування коду БЧХ (7, 3) з послідовним буферним регістром

Оскільки загалом можна сформувати 128 контрольних станів, то для подання помилкових конфігурацій при одиничних помилках, які можна виправити, з цього числа використовують 64 стани.

Для подання помилкових конфігурацій з подвійними (і більше) помилками потрібно, щоб число контрольних станів, що залишається, дорівнювало 64. Загальне число конфігурацій з подвійними помилками визначається так:

$$\binom{63}{2} = \frac{63!}{2!(63-2)!} = \frac{62 \cdot 63}{2} = 1953.$$

Спробуємо знайти моделі відстаней помилок для конфігурацій подвійних помилок. Всього одержуємо 31 модель для конфігурацій подвійних помилок, наприклад:

- модель відстаней помилок (1, 62);
- модель відстаней помилок (2, 61);
- модель відстаней помилок (3, 60);
- модель відстаней помилок (30, 33);
- модель відстаней помилок (31, 32).

За допомогою однієї моделі відстаней помилок можна виявити 63 конфігурації подвійних помилок, тому число конфігурацій подвійних помилок, які можуть бути виявлені за допомогою 31 моделі, — 63 · 31 = 1953. Таким чином, можуть бути виявлені всі конфігурації подвійних помилок.

Число контрольних станів, що залишилося, дорівнює 64. Немає необхідності у виправленні подвійних помилок, досить реєструвати їх наявність або відсутність. Незважаючи на те, що відбувається накладення моделей відстаней помилок, ніякі перешкоди не виникають. Щоб виправити подвійні помилки, необхідно ввести контрольні стани, що дозволяють виявляти відповідні конфігурації помилок. Для виявлення подвійних помилок можна скористатися будь-якою моделлю з 31 моделі відстаней для подвійних помилок. Наприклад, скористаємося моделлю відстаней помилок (1, 62).

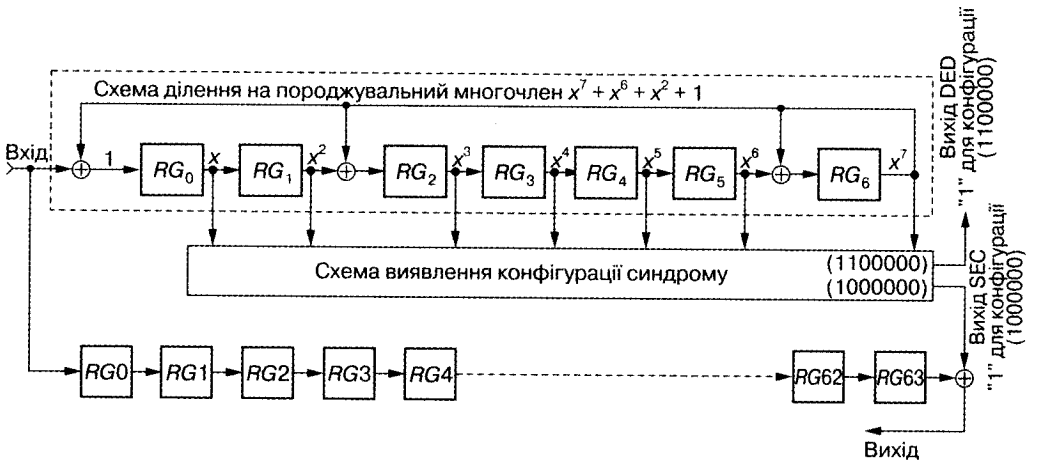


Рис. 32. Схема декодування коду БЧХ (63, 56) SEC

Залишком для моделі відстаней помилок (63) є $R(\alpha) = 1$, а залишком для моделі відстаней помилок (1, 62) — $R(\alpha) = \alpha + 1$. Це знаходиться в межах порядку породжувального многочлена. Тоді конфігурація синдрому, що визначає модель відстаней помилок (63) для виправлення одиничних помилок, має вигляд

$$(S\alpha_0, S\alpha_1, S\alpha_2, S\alpha_3, S\alpha_4, S\alpha_5, S\alpha_6) = (1000000).$$

Конфігурація синдрому, яка визначає модель відстаней помилок (1, 62) для виявлення подвійних помилок, подається так:

$$(S\alpha_0, S\alpha_1, S\alpha_2, S\alpha_3, S\alpha_4, S\alpha_5, S\alpha_6) = (1100000).$$

Після відшукання конфігурації синдрому схема декодування коду БЧХ (63, 56) SEC набуває вигляду, поданого на рис. 32, з якого бачимо, що приймальним буферним регістром є звичайний послідовний регістр.

18.1.2. Коди БЧХ DEC

У ч. 1 розглянуто питання декодування коду БЧХ (15, 7) DEC. Дослідимо декодування цього коду з використанням моделей відстаней помилок.

Як уже зауважувалося, код БЧХ (15, 7) — це код DEC, утворений з породжувального многочлена, коренями якого є послідовно зростаючі степені α, α^2, α^3, α^4 кореня α початкового многочлена. Для цього коду

- число кодових розрядів $n = 15$;
- число контрольних розрядів $m = 8$;
- число інформаційних розрядів $k = 7$;
- здатність до виправлення помилок — DEC (виправлення подвійних помилок);
- породжувальний многочлен — $x^8 + x^7 + x^6 + x^4 + 1$.

Схема кодування коду БЧХ (15, 7) наведена на рис. 33.

Структура коду БЧХ (15, 7) має такий вигляд:

Інформаційні розряди							Контрольні розряди							
a_6	a_5	a_4	a_3	a_2	a_1	a_0	c_7	c_6	c_5	c_4	c_3	c_2	c_1	c_0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Код має здатність виправляти подвійні помилки (DEC), тому у разі формування стандартної конфігурації такого коду як характерний базис можна вибрати всі конфігурації подвійних і одиничних помилок.

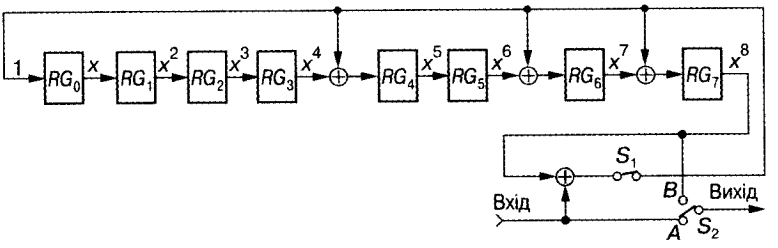


Рис. 33. Схема кодування коду БЧХ (15, 7)

Число контрольних станів, які можна подати за допомогою восьми контрольних розрядів, становить $2^8 = 256$. Число помилкових конфігурацій, які можуть бути виправлені за допомогою коду БЧХ (15, 7), можна подати у вигляді

- конфігурацій без помилок $\binom{15}{0}$;
- конфігурацій з одиночною помилкою $\binom{15}{1}$;
- конфігурацій з подвійною помилкою $\binom{15}{2}$.

Отже, загальне число помилкових конфігурацій, які можуть бути при цьому виправлені, визначається так:

$$\binom{15}{0}+\binom{15}{1}+\binom{15}{2}=\frac{15!}{0!(15-0)!}+\frac{15!}{1!(15-1)!}+\frac{15!}{2!(15-2)!}=1+15+105=121.$$

За допомогою контрольних розрядів можна відобразити 256 контрольних станів, а число помилкових конфігурацій, що виправляються, дорівнює 121. Отже, залишається ще 256 – 121 = 135 контрольних станів, які можуть бути використані. Оскільки уже вибрано конфігурації з одиничною та подвійною помилками, решта станів відводиться для помилкових конфігурацій з числом помилок більше ніж дві. Розраховуючи число конфігурацій з потрійною помилкою, отримуємо

$$\binom{15}{3}=\frac{15!}{3!(15-3)!}=\frac{13\cdot 14\cdot 15}{2\cdot 3}=455.$$

Отже, якщо навіть використовувати 135 контрольних станів, що залишилися для виявлення потрійних помилок, то можна буде виявити лише частину цих помилок. У кожному разі використанням коду БЧХ (15, 7) повністю виправляються всі помилкові конфігурації, аж до подвійних помилок включно.

Стандартна конфігурація коду БЧХ (15, 7) формується з використанням тих 121 конфігурацій, що піддаються виправленню як характерний базис (представницький базис). Число кодових розрядів дорівнює 15, тому має місце така помилкова послідовність:

$$e = (e_{14}, e_{13}, e_{12}, e_{11}, e_{10}, e_9, e_8, e_7, e_6, e_5, e_4, e_3, e_2, e_1, e_0),$$

у якій серед значень e_{14}—e_0 не більше ніж два символи дорівнюють "1". Визначимо всі моделі відстаней помилок аж до подвійних помилок.

Модель відстаней помилок для одиничних помилок має вигляд (15). У табл. 19 наведено ті конфігурації одиничних помилок, що піддаються виявленню.

Таблиця 19. Модель відстаней одиничних помилок і конфігурації виявлених помилок

Модель відстаней помилок	Конфігурації одиничних помилок
(15)	(e_0), (e_1), (e_2), (e_3), (e_4), (e_5), (e_6), (e_7), (e_8), (e_9), (e_{10}), (e_{11}), (e_{12}), (e_{13}), (e_{14})

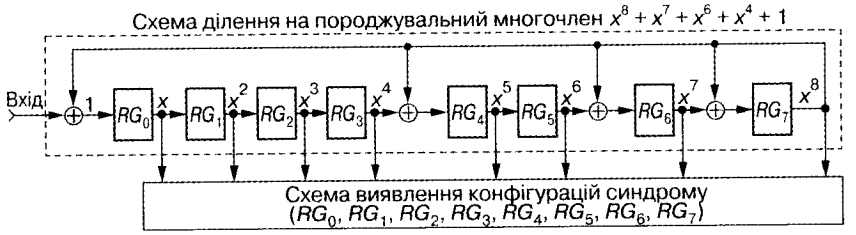
Оскільки α — один із коренів породжувального многочлена $x^8 + x^7 + x^6 + x^4 + 1$, то $\alpha^8 + \alpha^7 + \alpha^6 + \alpha^4 + 1 = 0$. У цьому разі виконується співвідношення

$$\alpha^8 = \alpha^7 + \alpha^6 + \alpha^4 + 1. \qquad (16)$$

Щоб знайти конфігурацію синдрому для визначення моделі відстаней помилок, згідно з залишком моделі відстаней помилок, необхідно за допомогою співвідношення (16) виразити залишок моделі многочленом, порядок якого не перевищує α^7. У разі коду БЧХ (15, 7) залишок моделі відстаней помилок не перевищує α^7. Отже, немає потреби використовувати співвідношення (16). Конфігурація синдрому для визначення моделі відстаней помилок наведена на рис. 34.

Знайшовши конфігурацію синдрому за допомогою поданої на рис. 34 схеми, можна побудувати схему декодування коду БЧХ (17, 7) (рис. 35).

Як приклад, розглянемо випадок помилок у четвертому розряді a_3 і у восьмому розряді c_7. Тоді конфігурація помилок має вигляд (e_{11}, e_7), тобто $e_{11} = 1$, $e_7 = 1$, решта розрядів дорівнюють "0". Цій помилковій конфігурації відповідає модель відстаней помилок (4, 11).



Модель відстаней помилок	Залишок моделей відстаней помилок	Конфігурація синдрому, що виявляє моделі відстаней помилок							
		RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7
(15)	$R(\alpha) = 1$	1	0	0	0	0	0	0	0
(1, 14)	$R(\alpha) = 1 + \alpha$	1	1	0	0	0	0	0	0
(2, 13)	$R(\alpha) = 1 + \alpha^2$	1	0	1	0	0	0	0	0
(3, 12)	$R(\alpha) = 1 + \alpha^3$	1	0	0	1	0	0	0	0
(4, 11)	$R(\alpha) = 1 + \alpha^4$	1	0	0	0	1	0	0	0
(5, 10)	$R(\alpha) = 1 + \alpha^5$	1	0	0	0	0	1	0	0
(6, 9)	$R(\alpha) = 1 + \alpha^6$	1	0	0	0	0	0	1	0
(7, 8)	$R(\alpha) = 1 + \alpha^7$	1	0	0	0	0	0	0	1

Рис. 34. Схема визначення конфігурації синдрому і взаємозв'язок моделей відстаней помилок і конфігурацій синдрому

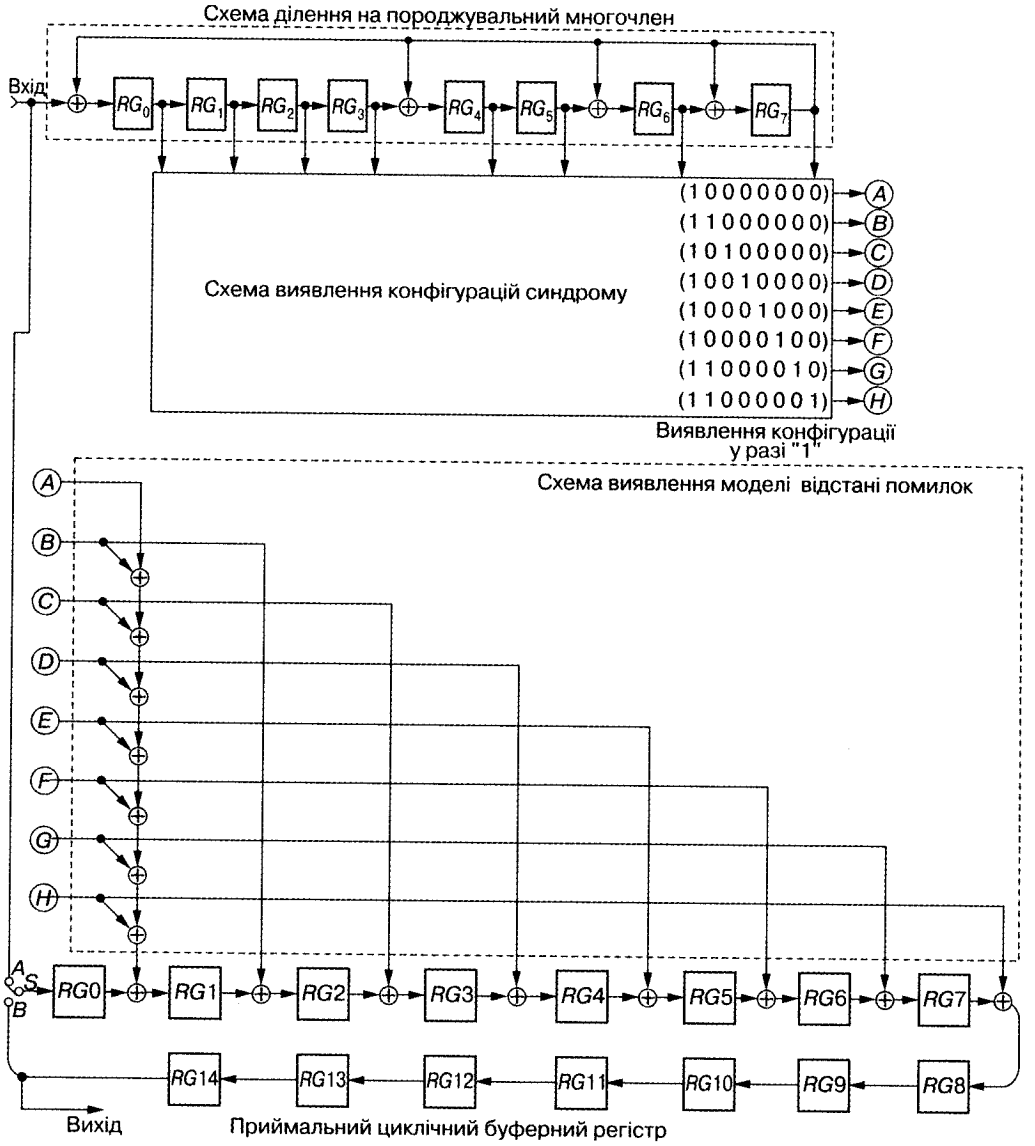


Рис. 35. Схема декодування моделей відстаней помилок для коду БЧХ (15, 7)

Згідно з рис. 34, конфігурація синдрому для виявлення моделі відстаней помилок (4, 11) має вигляд (10001000). Процес виправлення такої подвійної помилки наведено в табл. 20. У цій таблиці відображено операції з моменту закінчення введення всієї кодової послідовності, що приймається, тобто з моменту відшукання наступного залишку від ділення кодової послідовності, що приймається, на породжувальний многочлен:

$$R(\alpha) = e_{14}\alpha^{14} + e_{13}\alpha^{13} + e_{12}\alpha^{12} + (e_{11})\alpha^{11} +$$
$$+ e_{10}\alpha^{10} + e_9\alpha^9 + e_8\alpha^8 + (e_7)\alpha^7 + e_6\alpha^6 +$$
$$+ e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0, \qquad (17)$$

де (e_7), (e_{11}) дорівнюють "1", а решта значень — "0".

Порядок породжувального многочлена дорівнює 8, тому дійсно шуканий залишок виражається за допомогою співвідношення (16) многочленом, порядок якого не перевищує α^7. Іншими словами, підставляючи у співвідношення (17) дійсний залишок

$$\alpha^8 = \alpha^7 + \alpha^6 + \alpha^4 + 1,$$
$$\alpha^9 = \alpha \cdot \alpha^8 = \alpha^8 + \alpha^7 + \alpha^5 + \alpha =$$
$$= (\alpha^7 + \alpha^6 + \alpha^4 + 1) + \alpha^7 + \alpha^5 + \alpha =$$
$$= \alpha^6 + \alpha^5 + \alpha^4 + \alpha + 1,$$
$$\alpha^{10} = \alpha \cdot \alpha^9 = \alpha^7 + \alpha^6 + \alpha^5 + \alpha^2 + \alpha,$$
$$\alpha^{11} = \alpha \cdot \alpha^{10} = \alpha^8 + \alpha^7 + \alpha^6 + \alpha^3 + \alpha^2 =$$
$$= (\alpha^7 + \alpha^6 + \alpha^4 + 1) + \alpha^7 + \alpha^6 + \alpha^3 + \alpha^2 =$$
$$= \alpha^4 + \alpha^3 + \alpha^2 + 1,$$
$$\alpha^{12} = \alpha \cdot \alpha^{11} = \alpha^5 + \alpha^4 + \alpha^3 + \alpha,$$
$$\alpha^{13} = \alpha \cdot \alpha^{12} = \alpha^6 + \alpha^5 + \alpha^4 + \alpha^2,$$
$$\alpha^{14} = \alpha \cdot \alpha^{13} = \alpha^7 + \alpha^6 + \alpha^5 + \alpha^3,$$

одержуємо

$$R(\alpha) = (e_{14} + e_{10} + e_8 + (e_7))\alpha^7 +$$
$$+ (e_{14} + e_{13} + e_{10} + e_9 + e_8 + e_6)\alpha^6 +$$
$$+ (e_{14} + e_{13} + e_{12} + e_{10} + e_9 + e_5)\alpha^5 +$$
$$+ (e_{13} + e_{12} + (e_{11}) + e_9 + e_8 + e_4)\alpha^4 +$$
$$+ (e_{14} + e_{12} + (e_{11}) + e_3)\alpha^3 +$$
$$+ (e_{13} + e_{11} + e_{10} + e_2)\alpha^2 +$$
$$+ (e_{12} + e_{10} + e_9 + e_1)\alpha + ((e_{11}) + e_9 + e_8 + e_0). \qquad (18)$$

Таким чином,

$$R(\alpha) = (e_7)\alpha^7 + (e_{11})\alpha^4 + (e_{11})\alpha^3 + (e_{11})\alpha^2 + (e_{11}).$$

Таблиця 20. Процес виправлення

Перемикач	Такт	Ділення на породжувальний многочлен								Вхід схеми виявлення конфігурації синдрому
		RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7	
	0	e_{11}	0	e_{11}	e_{11}	e_{11}	0	0	e_7	(10111001)
	1	e_7	e_{11}	0	e_{11}	$e_{11}+e_7$	e_{11}	e_7	e_7	(11010111)
	2	e_7	e_7	e_{11}	0	$e_{11}+e_7$	$e_{11}+e_7$	$e_{11}+e_7$	0	(11100000)
	3	0	e_7	e_7	e_{11}	0	$e_{11}+e_7$	$e_{11}+e_7$	$e_{11}+e_7$	(01110000)
	4	$e_{11}+e_7$	0	e_7	e_7	e_7	0	0	0	(00111000)
	5	0	$e_{11}+e_7$	0	e_7	e_7	e_7	0	0	(00011100)
	6	0	0	$e_{11}+e_7$	0	e_7	e_7	e_7	0	(00001110)
	7	0	0	0	$e_{11}+e_7$	0	e_7	e_7	e_7	(00000111)
B	8	e_7	0	0	0	e_{11}	0	0	0	(10001000) Конфігурація синдрому для моделі відстаней помилок (4, 11) → Вихід виявленої конфігурації "1"
	9	0	e_7	0	0	0	e_{11}	0	0	(01000100)
	10	0	0	e_7	0	0	0	e_{11}	0	(00100010)
	11	0	0	0	e_7	0	0	0	e_{11}	(00010001)
	12	e_{11}	0	0	0	$e_{11}+e_7$	0	e_{11}	e_{11}	(10000011)
	13	e_{11}	e_{11}	0	0	e_{11}	$e_{11}+e_7$	e_{11}	0	(11001010)
	14	0	e_{11}	e_{11}	0	0	e_{11}	$e_{11}+e_7$	e_{11}	(01100101)
	15	—	—	—	—	—	—	—	—	—
	16	—	—	—	—	—	—	—	—	—
	17	—	—	—	—	—	—	—	—	—
	18	—	—	—	—	—	—	—	—	—
	19	—	—	—	—	—	—	—	—	—
	20	—	—	—	—	—	—	—	—	—
	21	—	—	—	—	—	—	—	—	—
A	22	—	—	—	—	—	—	—	—	—
	23	—	—	—	—	—	—	—	—	—
	24	—	—	—	—	—	—	—	—	—
	25	—	—	—	—	—	—	—	—	—
	26	—	—	—	—	—	—	—	—	—
	27	—	—	—	—	—	—	—	—	—
	28	—	—	—	—	—	—	—	—	—
	29	—	—	—	—	—	—	—	—	—

e_{11}, $e_7 = 1$, інші дорівнюють "0";

подвійних помилок

Приймальний циклічний буферний регістр															Вихід
*RG*0	*RG*1	*RG*2	*RG*3	*RG*4	*RG*5	*RG*6	*RG*7	*RG*8	*RG*9	*RG*10	*RG*11	*RG*12	*RG*13	*RG*14	
c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	a_1	a_2	$\overline{a}_3$	a_4	a_5	a_6	—
a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	a_1	a_2	$\overline{a}_3$	a_4	a_5	—
a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	a_1	a_2	$\overline{a}_3$	a_4	—
a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	a_1	a_2	$\overline{a}_3$	—
$\overline{a}_3$	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	a_1	a_2	—
a_2	$\overline{a}_3$	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	a_1	—
a_1	a_2	$\overline{a}_3$	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	a_0	—
a_0	a_1	a_2	$\overline{a}_3$	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\overline{c}_7$	—
$\overline{c}_7$	a_0	a_1	a_2	$\overline{a}_3$	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	c_6	—
→ Вихід схеми виявлення моделей відстаней помилок															
1	(0)	(0)	(0)	1	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	
c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	c_5	—
c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	c_0	c_1	c_2	c_3	c_4	—
c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	c_0	c_1	c_2	c_3	—
c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	c_0	c_1	c_2	—
c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	c_0	c_1	—
c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	c_0	—
c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_6	a_6
—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_5	a_5
—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_4	a_4
—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_3	a_3
—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_2	a_2
—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_1	a_1
—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	a_0	a_0
—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_7
—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_6
—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_5
—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_4
—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_3
—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_2
—	—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_1
—	—	—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_0

$\overline{a}_3 = a_3 + e_{11}$, $\overline{c}_7 = c_7 + e_7$.

18.1.3. БЧХ коди ТЕС

Коди БЧХ (15, 5) ТЕС використовуються в цифровій системі магнітного запису інформації і утворюються з породжувального многочлена $x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$, корені якого α, α^2, α^3, α^4, α^5, α^6 є результатами піднесення до степеня числа α, що є коренем вихідного (початкового) многочлена $x^4 + x + 1$. Можливість виправляти помилки коду БЧХ (15, 5) ТЕС зумовлюють його параметри:

- число розрядів коду $n = 15$;
- число контрольних розрядів $m = 10$;
- число інформаційних розрядів $k = 15$.

Структура коду БЧХ (15, 5) має такий вигляд:

Інформаційні розряди					Контрольні розряди									
a_4	a_3	a_2	a_1	a_0	c_9	c_8	c_7	c_6	c_5	c_4	c_3	c_2	c_1	c_0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Порядок і схема, що здійснює формування кодів БЧХ (15, 5), наведені на рис. 36.

Контрольні розряди формуються з інформаційних розрядів таким чином:

$$c_9 = a_4 + a_3 + a_1 \pmod 2,$$
$$c_8 = a_3 + a_2 + a_0 \pmod 2,$$
$$c_7 = a_4 + a_3 + a_2 \pmod 2,$$
$$c_6 = a_3 + a_2 + a_1 \pmod 2,$$
$$c_5 = a_2 + a_1 + a_0 \pmod 2,$$
$$c_4 = a_4 + a_3 + a_0 \pmod 2,$$
$$c_3 = a_4 + a_2 + a_1 \pmod 2,$$
$$c_2 = a_3 + a_1 + a_0 \pmod 2,$$
$$c_1 = a_4 + a_3 + a_2 + a_1 + a_0 \pmod 2,$$
$$c_0 = a_4 + a_2 + a_0 \pmod 2.$$

Обернена залежність між контрольними і інформаційними розрядами наведена в табл. 21.

Таблиця 21. Інформаційні і контрольні розряди, що їх відображають

Інформаційний розряд	Контрольні розряди	Число розрядів, що змінюються, у разі зміни інформаційного розряду
a_4	c_9, c_7, c_4, c_3, c_1, c_0	7
a_3	c_9, c_8, c_7, c_6, c_4, c_2, c_3	8
a_2	c_8, c_7, c_6, c_5, c_3, c_1,c_0	8
a_1	c_9, c_6, c_5, c_3, c_2, c_1	7
a_0	c_8, c_5, c_4, c_2, c_1, c_0	7

Як бачимо з цієї таблиці, інформаційні розряди коду БЧХ (15, 5) розрізняються не менше ніж сімома контрольними розрядами. Отже, можливість виправлення помилок у цьому коді існує.

Створюючи стандартний масив кодів БЧХ, керуються загальноприйнятим припущенням, а саме: помилка можлива не більше ніж в трьох розрядах коду.

Число контрольних розрядів коду БЧХ (15, 5) дорівнює 10. Число можливих комбінацій, що відображаються цими розрядами, становить $2^{10} = 1024$. Число помилкових комбінацій, яке можливе за наявності помилок не більше ніж в трьох розрядах коду, наступне:

- помилок немає — $\left(\frac{15}{0}\right)$;
- помилка в одному розряді — $\left(\frac{15}{1}\right)$;
- помилки в двох розрядах — $\left(\frac{15}{2}\right)$;
- помилки в трьох розрядах — $\left(\frac{15}{3}\right)$.

Отже, загальне число помилкових комбінацій, які можуть бути виправлені за допомогою даних кодів, визначається так:

$$\left(\frac{15}{0}\right)+\left(\frac{15}{1}\right)+\left(\frac{15}{2}\right)+\left(\frac{15}{3}\right)=1+15+105+455=576,$$

тобто, десяти контрольних розрядів цілком достатньо для виправлення усіх помилок, включаючи і помилки у трьох розрядах.

Десятьма розрядами можна контролювати 1024 кодових комбінації, отже, надмірне число контрольованих кодових комбінацій становить 1024 – 576 = = 448. Таким чином, хоча і передбачається наявність не більше ніж трьох помилок, проте 448 комбінацій, що залишилися, дозволяють виправляти і чотири помилки. Однак число помилкових комбінацій при помилках в чотирьох розрядах коду становить

$$\left(\frac{15}{4}\right)=\frac{15!}{4!(15-4)!}=\frac{12\cdot13\cdot14\cdot15}{1\cdot2\cdot3\cdot4}=1365,$$

тому 448 комбінаціями можна виправити лише частину кодів з помилками в чотирьох розрядах. Отже, код БЧХ (15, 5) усе-таки розрахований на виправлення не більше ніж трьох помилкових розрядів.

У разі виправлення помилок шляхом декодування кодових конфігурацій схема декодування просто визначає помилкові кодові конфігурації. Розглянемо такий процес за наявності помилок не більше ніж в трьох розрядах.

Кодові конфігурації з помилками в одному або двох розрядах визначаються унаслідок декодування конфігурацій коду БЧХ (15, 7). Оскільки число розрядів

① Інформаційна послідовність
$a = (a_4, a_3, a_2, a_1, a_0)$

② Породжувальний многочлен
$G(x) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1$

③ $A(x)\cdot x^{10} \,/\, G(x) = R(x)$

→ Інформаційний многочлен
$A(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x^1 + a_0$

→ $A(x)\cdot x^{10}$ (степінь породжувального многочлена)
$A(x)\cdot x^{10} = a_4x^{14} + a_3x^{13} + a_2x^{12} + a_1x^{11} + a_0x^{10}$

$$
\begin{array}{l}
x^{10}+x^8+x^5+x^4+x^2+x+1) \;\; \dfrac{a_4x^4 + a_3x^3 + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^2 + \left[\begin{smallmatrix} a_3 \\ + \\ a_1 \end{smallmatrix}\right]x + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{smallmatrix}\right]}{a_4x^{14} + a_3x^{13} \quad + a_2x^{12} \quad + a_1x^{11} \quad + a_0x^{10}} \\
\dfrac{a_{14}x^{14} \quad + a_4x^{12} \quad + a_4x^9 \quad + a_4x^8 \quad + a_4x^6 \quad + a_4x^5 \quad + a_4x^4}{a_3x^{13} + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^{12} \quad + a_1x^{11} \quad + a_0x^{10} + a_4x^9 \quad + a_4x^8 \quad + a_4x^6 \quad + a_4x^5 \quad + a_4x^4} \\
\dfrac{a_3x^{13} \quad + a_3x^{11} \quad + a_3x^8 \quad + a_3x^7 \quad + a_3x^5 \quad + a_3x^4 \quad + a_3x^3}{\left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^{12} + \left[\begin{smallmatrix} a_3 \\ + \\ a_1 \end{smallmatrix}\right]x^{11} + a_0x^{10} + a_4x^9 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \end{smallmatrix}\right]x^8 + a_3x^7 + a_4x^6 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \end{smallmatrix}\right]x^5 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \end{smallmatrix}\right]x^4 + a_3x^3} \\
\dfrac{\left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^{12} \quad + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^{10} \quad + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^7 + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^6 \quad + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^4 + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^3 + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^2}{\left[\begin{smallmatrix} a_3 \\ + \\ a_1 \end{smallmatrix}\right]x^{11} + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{smallmatrix}\right]x^{10} + a_4x^9 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \end{smallmatrix}\right]x^8 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \end{smallmatrix}\right]x^7 + a_4x^6 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \end{smallmatrix}\right]x^5 + \left[\begin{smallmatrix} a_3 \\ + \\ a_2 \end{smallmatrix}\right]x^4 + \left[\begin{smallmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \end{smallmatrix}\right]x^3 + \left[\begin{smallmatrix} a_4 \\ + \\ a_2 \end{smallmatrix}\right]x^2 \longrightarrow}
\end{array}
$$

$$\begin{array}{r}
\begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x^{11} + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x^9 + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x^6 + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x^5 + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x^3 + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x^2 + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x \\
\hline
\begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^{10} + \begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \end{bmatrix} x^9 + \begin{bmatrix} a_4 \\ + \\ a_3 \end{bmatrix} x^8 + \begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \end{bmatrix} x^7 + \begin{bmatrix} a_3 \\ + \\ a_2 \\ + \\ a_1 \end{bmatrix} x^6 + \begin{bmatrix} a_4 \\ + \\ a_3 \end{bmatrix} x^5 + \begin{bmatrix} a_3 \\ + \\ a_2 \end{bmatrix} x^4 + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^3 + \begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \\ + \\ a_1 \end{bmatrix} x^2 + \begin{bmatrix} a_3 \\ + \\ a_1 \end{bmatrix} x \\
\hline
\begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^{10} + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^8 + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^5 + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^4 + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x^2 + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} x + \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} \\
\hline
\begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_1 \end{bmatrix} \begin{bmatrix} a_3 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} \begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \end{bmatrix} \begin{bmatrix} a_3 \\ + \\ a_2 \\ + \\ a_1 \end{bmatrix} \begin{bmatrix} a_2 \\ + \\ a_1 \\ + \\ a_0 \end{bmatrix} \begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \end{bmatrix} \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix} \begin{bmatrix} a_3 \\ + \\ a_2 \\ + \\ a_1 \end{bmatrix} \begin{bmatrix} a_4 \\ + \\ a_3 \\ + \\ a_2 \\ + \\ a_1 \\ + \\ a_0 \end{bmatrix} \begin{bmatrix} a_4 \\ + \\ a_2 \\ + \\ a_0 \end{bmatrix}
\end{array}$$

$\longrightarrow R(x) = c_9x^9 + c_8x^8 + c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$

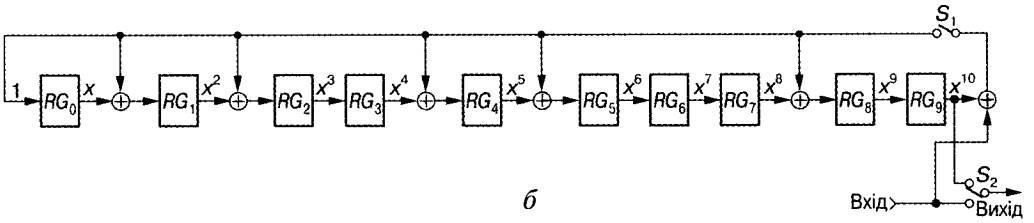
$c_9 = a_4 + a_3 + a_1 \pmod 2$
$c_8 = a_3 + a_2 + a_0 \pmod 2$
$c_7 = a_4 + a_3 + a_2 \pmod 2$
$c_6 = a_3 + a_2 + a_1 \pmod 2$
$c_5 = a_2 + a_1 + a_0 \pmod 2$
$c_4 = a_4 + a_3 + a_0 \pmod 2$
$c_3 = a_4 + a_2 + a_1 \pmod 2$
$c_2 = a_3 + a_1 + a_0 \pmod 2$
$c_1 = a_4 + a_3 + a_2 + a_1 + a_0 \pmod 2$
$c_0 = a_4 + a_2 + a_0 \pmod 2$

④ $A(x)\cdot x^{10} + R(x)$

⑤ Многочлен коду $v(x)$

$\longrightarrow$ Многочлен коду $v(x) = a_4x^{14} + a_3x^{13} + a_2x^{12} + a_1x^{11} + a_0x^{10} + c_9x^9 + c_8x^8 + c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$

$\longrightarrow$ Розряди кодової послідовності
$v = (a_4, a_3, a_2, a_1, a_0, c_9, c_8, c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0)$



б

Рис. 36. Порядок (*а*) і схема кодування кодів БЧХ (15, 5) (*б*)

кодів БЧХ (15, 7) і БЧХ (15, 5) однакове, то і кодові конфігурації для них теж однакові. Проте оскільки для цих кодів породжувальні многочлени різні, то і конфігурації синдромів для визначення шуканих кодових конфігурацій також розрізняються.

Моделі відстаней помилок у разі однієї або двох помилок наступні:

- модель відстаней помилок (15);
- модель відстаней помилок (1, 14);
- модель відстаней помилок (2, 13);
- модель відстаней помилок (3, 12);
- модель відстаней помилок (4, 11);
- модель відстаней помилок (5, 10);
- модель відстаней помилок (6, 9);
- модель відстаней помилок (7, 8).

У разі помилок в трьох розрядах моделі відстаней помилок мають вигляд (i, j, k). Визначення всіх моделей відстаней помилок, що задовольняють умову $i + j + k = 15$; $i \le j$, k; $j < k$ (випадок $i = j = k$ виключається), означає їх визначення при помилках у трьох розрядах. Ці моделі можна згрупувати таким чином:

1. Для $i = 1$:

(1, 1, 13), (1, 2, 12), (1, 3, 11),
(1, 4, 10), (1, 5, 9), (1, 6, 8),
(1, 7, 7), (1, 8, 6), (1, 9, 5),
(1, 10, 4), (1, 11, 3), (1, 12, 2).

2. Для $i = 2$:

(2, 2, 11), (2, 3, 10), (2, 4, 9),
(2, 5, 8), (2, 6, 7), (2, 7, 6),
(2, 8, 5), (2, 9, 4), (2, 10, 3).

3. Для $i = 3$:

(3, 3, 9), (3, 4, 8), (3, 5, 7),
(3, 6, 6), (3, 7, 5), (3, 8, 4).

4. Для $i = 4$:

(4, 4, 7), (4, 5, 6), (4, 6, 5).

5. Для $i = 5$:

(5, 5, 5).

Загальне число моделей відстаней помилок становить 31. Отже, число моделей з помилками у трьох розрядах теж дорівнює 31. Моделі відстаней помилок і конфігурації виявних помилок у трьох розрядах наведені в табл. 22.

Таблиця 22. Моделі відстаней помилок і конфігурації потрійних помилок

Модель відстаней помилок	Конфігурації виявних потрійних помилок				
(1, 1, 13)	(e_2, e_1, e_0), (e_7, e_6, e_5), (e_{12}, e_{11}, e_{10}),	(e_3, e_2, e_1), (e_8, e_7, e_6), (e_{13}, e_{12}, e_{11}),	(e_4, e_3, e_2), (e_9, e_8, e_7), (e_{14}, e_{13}, e_{12}),	(e_5, e_4, e_3), (e_{10}, e_9, e_8), (e_{14}, e_{13}, e_0),	(e_6, e_5, e_4), (e_{11}, e_{10}, e_9), (e_{14}, e_1, e_0),
(1, 2, 12)	(e_3, e_1, e_0), (e_9, e_6, e_5), (e_{13}, e_{11}, e_{10}),	(e_4, e_2, e_1), (e_9, e_7, e_6), (e_{14}, e_{12}, e_{11}),	(e_5, e_3, e_2), (e_{10}, e_8, e_7), (e_{13}, e_{12}, e_0),	(e_6, e_4, e_3), (e_{11}, e_9, e_8), (e_{14}, e_{13}, e_1),	(e_7, e_5, e_4), (e_{12}, e_{10}, e_9), (e_{14}, e_2, e_0),
(1, 3, 11)	(e_4, e_1, e_0), (e_9, e_6, e_5), (e_{14}, e_{11}, e_{10}),	(e_5, e_2, e_1), (e_{10}, e_7, e_6), (e_{12}, e_{11}, e_0),	(e_6, e_3, e_2), (e_{11}, e_9, e_7), (e_{13}, e_{12}, e_1),	(e_7, e_4, e_3), (e_{12}, e_9, e_8), (e_{14}, e_{13}, e_2),	(e_8, e_5, e_4), (e_{13}, e_{10}, e_9), (e_{14}, e_3, e_0),
(1, 4, 10)	(e_5, e_1, e_0), (e_{10}, e_6, e_5), (e_{11}, e_{10}, e_1),	(e_6, e_2, e_1), (e_{11}, e_7, e_6), (e_{12}, e_{11}, e_2),	(e_7, e_3, e_2), (e_{12}, e_9, e_7), (e_{13}, e_{12}, e_3),	(e_9, e_4, e_3), (e_{13}, e_9, e_8), (e_{14}, e_{13}, e_4),	(e_9, e_5, e_4), (e_{14}, e_{10}, e_9), (e_{14}, e_5, e_0),
(1, 5, 9)	(e_6, e_1, e_0), (e_{11}, e_6, e_5), (e_{11}, e_{10}, e_1),	(e_7, e_2, e_1), (e_{12}, e_7, e_6), (e_{12}, e_{11}, e_2),	(e_8, e_3, e_2), (e_{13}, e_8, e_7), (e_{13}, e_{12}, e_3),	(e_9, e_4, e_3), (e_{14}, e_9, e_8), (e_{14}, e_{13}, e_4),	(e_{10}, e_5, e_4), (e_{10}, e_9, e_0), (e_{14}, e_5, e_0),
(1, 6, 8)	(e_7, e_1, e_0), (e_{12}, e_6, e_5), (e_{11}, e_{10}, e_2),	(e_8, e_2, e_1), (e_{13}, e_7, e_6), (e_{12}, e_{11}, e_3),	(e_9, e_3, e_2), (e_{14}, e_8, e_7), (e_{14}, e_{12}, e_4),	(e_{10}, e_4, e_3), (e_9, e_8, e_0), (e_{14}, e_{13}, e_5),	(e_{11}, e_5, e_4), (e_{10}, e_9, e_1), (e_{14}, e_6, e_0),
(1, 7, 7)	(e_8, e_1, e_0), (e_{13}, e_6, e_5), (e_{11}, e_{10}, e_3),	(e_9, e_2, e_1), (e_{14}, e_7, e_6), (e_{12}, e_{11}, e_4),	(e_{10}, e_3, e_2), (e_9, e_7, e_0), (e_{13}, e_{12}, e_5),	(e_{11}, e_4, e_3), (e_9, e_8, e_1), (e_{14}, e_{13}, e_6),	(e_{12}, e_5, e_4), (e_{10}, e_9, e_2), (e_{14}, e_7, e_0),
(1, 8, 6)	(e_9, e_1, e_0), (e_{14}, e_6, e_5), (e_{11}, e_{10}, e_4),	(e_{10}, e_2, e_1), (e_7, e_6, e_0), (e_{12}, e_{11}, e_5),	(e_{11}, e_3, e_2), (e_8, e_7, e_1), (e_{13}, e_{12}, e_6),	(e_{12}, e_4, e_3), (e_9, e_8, e_2), (e_{14}, e_{12}, e_7),	(e_{13}, e_5, e_4), (e_{10}, e_9, e_2), (e_{14}, e_9, e_0),
(1, 9, 5)	(e_{10}, e_1, e_0), (e_6, e_5, e_0), (e_{11}, e_{10}, e_5),	(e_{11}, e_2, e_1), (e_7, e_6, e_1), (e_{12}, e_{11}, e_6),	(e_{12}, e_3, e_2), (e_9, e_7, e_2), (e_{13}, e_{12}, e_7),	(e_{12}, e_4, e_3), (e_9, e_8, e_2), (e_{14}, e_{13}, e_8),	(e_{14}, e_5, e_4), (e_{10}, e_9, e_4), (e_{14}, e_9, e_0),
(1, 10, 4)	(e_{11}, e_1, e_0), (e_6, e_5, e_1), (e_{11}, e_{10}, e_6),	(e_{12}, e_2, e_1), (e_7, e_6, e_2), (e_{12}, e_{11}, e_7),	(e_{13}, e_3, e_2), (e_9, e_7, e_3), (e_{13}, e_{12}, e_8),	(e_{14}, e_4, e_3), (e_9, e_8, e_4), (e_{14}, e_{13}, e_9),	(e_5, e_4, e_0), (e_{10}, e_9, e_5), (e_{14}, e_{11}, e_0),
(1, 11, 3)	(e_{12}, e_1, e_0), (e_6, e_5, e_2), (e_{11}, e_{10}, e_7),	(e_{13}, e_2, e_1), (e_7, e_6, e_3), (e_{12}, e_{11}, e_8),	(e_{14}, e_3, e_2), (e_8, e_7, e_4), (e_{13}, e_{12}, e_9),	(e_4, e_3, e_0), (e_9, e_8, e_5), (e_{14}, e_{13}, e_{10}),	(e_5, e_4, e_1), (e_{10}, e_9, e_6), (e_{14}, e_{11}, e_0),
(1, 12, 2)	(e_{13}, e_1, e_0), (e_6, e_5, e_3), (e_{11}, e_{10}, e_8),	(e_{14}, e_2, e_1), (e_7, e_6, e_4), (e_{12}, e_{11}, e_9),	(e_3, e_2, e_0), (e_8, e_7, e_5), (e_{13}, e_{12}, e_{10}),	(e_4, e_4, e_3), (e_9, e_8, e_6), (e_{14}, e_{13}, e_{11}),	(e_5, e_4, e_2), (e_{10}, e_9, e_7), (e_{14}, e_7, e_0),

Продовження табл. 22

Модель відстаней помилок	Конфігурації виявних потрійних помилок				
(2, 2, 11)	(e_4, e_2, e_0), (e_9, e_7, e_5), (e_{14}, e_{12}, e_{10}),	(e_5, e_3, e_1), (e_{10}, e_9, e_6), (e_{13}, e_{11}, e_0),	(e_6, e_4, e_2), (e_{11}, e_9, e_7), (e_{14}, e_{12}, e_1),	(e_7, e_5, e_3), (e_{12}, e_{10}, e_9), (e_{12}, e_2, e_0),	(e_9, e_6, e_4), (e_{13}, e_{11}, e_9), (e_{14}, e_3, e_1),
(2, 3, 10)	(e_5, e_2, e_0), (e_{10}, e_7, e_5), (e_{12}, e_{10}, e_0),	(e_6, e_3, e_1), (e_{11}, e_9, e_6), (e_{13}, e_{11}, e_1),	(e_7, e_4, e_2), (e_{12}, e_9, e_7), (e_{14}, e_{12}, e_2),	(e_8, e_5, e_3), (e_{13}, e_{10}, e_8), (e_{13}, e_3, e_0),	(e_9, e_6, e_4), (e_{14}, e_{11}, e_9), (e_{14}, e_4, e_1),
(2, 4, 9)	(e_6, e_2, e_0), (e_{11}, e_7, e_5), (e_{12}, e_{10}, e_1),	(e_7, e_3, e_1), (e_{12}, e_8, e_6), (e_{13}, e_{11}, e_2),	(e_9, e_4, e_2), (e_{13}, e_9, e_7), (e_{14}, e_{12}, e_3),	(e_9, e_5, e_3), (e_{14}, e_{10}, e_8), (e_{13}, e_4, e_0),	(e_{10}, e_6, e_4), (e_{11}, e_9, e_0), (e_{14}, e_5, e_1),
(2, 5, 8)	(e_7, e_2, e_0), (e_{12}, e_7, e_5), (e_{12}, e_{10}, e_2),	(e_8, e_3, e_1), (e_{13}, e_8, e_6), (e_{13}, e_{11}, e_3),	(e_9, e_4, e_2), (e_{14}, e_9, e_7), (e_{14}, e_{12}, e_4),	(e_{10}, e_5, e_3), (e_{10}, e_8, e_0), (e_{13}, e_5, e_0),	(e_{11}, e_6, e_4), (e_{11}, e_9, e_1), (e_{14}, e_6, e_1),
(2, 6, 7)	(e_8, e_2, e_0), (e_{13}, e_7, e_5), (e_{12}, e_{10}, e_3),	(e_9, e_3, e_1), (e_{14}, e_9, e_6), (e_{13}, e_{11}, e_4),	(e_{10}, e_4, e_2), (e_9, e_7, e_0), (e_{14}, e_{12}, e_5),	(e_{11}, e_5, e_3), (e_{10}, e_8, e_1), (e_{13}, e_6, e_0),	(e_{12}, e_6, e_4), (e_{11}, e_9, e_1), (e_{14}, e_7, e_1),
(2, 7, 6)	(e_9, e_2, e_0), (e_{14}, e_7, e_5), (e_{12}, e_{10}, e_4),	(e_{10}, e_3, e_1), (e_9, e_6, e_0), (e_{13}, e_{11}, e_5),	(e_{11}, e_4, e_2), (e_9, e_7, e_1), (e_{14}, e_{12}, e_6),	(e_{12}, e_5, e_3), (e_{10}, e_8, e_2), (e_{13}, e_7, e_0),	(e_{13}, e_6, e_4), (e_{11}, e_9, e_3), (e_{14}, e_8, e_1),
(2, 8, 5)	(e_{10}, e_2, e_0), (e_7, e_5, e_0), (e_{12}, e_{10}, e_5),	(e_{11}, e_3, e_1), (e_8, e_6, e_1), (e_{13}, e_{11}, e_6),	(e_{12}, e_4, e_2), (e_9, e_7, e_2), (e_{14}, e_{12}, e_7),	(e_{13}, e_5, e_3), (e_{10}, e_8, e_3), (e_{13}, e_8, e_0),	(e_{14}, e_6, e_4), (e_{11}, e_9, e_4), (e_{14}, e_9, e_1),
(2, 9, 4)	(e_{11}, e_2, e_0), (e_7, e_5, e_1), (e_{12}, e_{10}, e_6),	(e_{12}, e_3, e_1), (e_8, e_6, e_2), (e_{13}, e_{11}, e_7),	(e_{13}, e_4, e_2), (e_9, e_7, e_3), (e_{14}, e_{12}, e_8),	(e_{14}, e_5, e_3), (e_{10}, e_8, e_4), (e_{13}, e_9, e_0),	(e_6, e_4, e_0), (e_{11}, e_9, e_5), (e_{14}, e_{10}, e_1),
(2, 10, 3)	(e_{12}, e_2, e_0), (e_7, e_5, e_2), (e_{12}, e_{10}, e_7),	(e_{13}, e_3, e_1), (e_8, e_6, e_3), (e_{13}, e_{11}, e_8),	(e_{14}, e_4, e_2), (e_9, e_7, e_4), (e_{14}, e_{12}, e_9),	(e_5, e_3, e_0), (e_{10}, e_8, e_5), (e_{13}, e_{10}, e_0),	(e_6, e_4, e_1), (e_{11}, e_9, e_5), (e_{14}, e_{11}, e_1),
(3, 3, 9)	(e_6, e_3, e_0), (e_{11}, e_9, e_5), (e_{12}, e_{10}, e_1),	(e_7, e_4, e_1), (e_{12}, e_9, e_6), (e_{14}, e_{11}, e_2),	(e_8, e_5, e_2), (e_{13}, e_{10}, e_7), (e_{12}, e_3, e_0),	(e_9, e_6, e_3), (e_{14}, e_{11}, e_8), (e_{13}, e_4, e_1),	(e_{10}, e_7, e_4), (e_{12}, e_9, e_0), (e_{14}, e_5, e_2),
(3, 4, 8)	(e_7, e_3, e_0), (e_{12}, e_9, e_5), (e_{13}, e_{10}, e_2),	(e_8, e_4, e_1), (e_{13}, e_9, e_6), (e_{14}, e_{11}, e_3),	(e_9, e_5, e_2), (e_{14}, e_{10}, e_7), (e_{12}, e_4, e_0),	(e_{10}, e_6, e_3), (e_{11}, e_8, e_0), (e_{13}, e_5, e_1),	(e_{11}, e_7, e_4), (e_{12}, e_9, e_1), (e_{14}, e_6, e_2),
(3, 5, 7)	(e_8, e_3, e_0), (e_{13}, e_9, e_5), (e_{13}, e_{10}, e_3),	(e_9, e_4, e_1), (e_{14}, e_9, e_6), (e_{14}, e_{11}, e_4),	(e_{10}, e_5, e_2), (e_{10}, e_7, e_0), (e_{12}, e_5, e_0),	(e_{11}, e_6, e_3), (e_{11}, e_9, e_0), (e_{13}, e_6, e_1),	(e_{12}, e_7, e_4), (e_{12}, e_9, e_2), (e_{14}, e_7, e_2),
(3, 6, 6)	(e_9, e_3, e_0), (e_{14}, e_9, e_5), (e_{13}, e_{10}, e_4),	(e_{10}, e_4, e_1), (e_9, e_6, e_0), (e_{14}, e_{11}, e_5),	(e_{11}, e_5, e_2), (e_{10}, e_7, e_1), (e_{12}, e_6, e_0),	(e_{12}, e_6, e_3), (e_{11}, e_8, e_2), (e_{13}, e_7, e_1),	(e_{13}, e_7, e_4), (e_{12}, e_9, e_3), (e_{14}, e_9, e_2),

Закінчення табл. 22

Модель відстаней помилок	Конфігурації виявних потрійних помилок				
(3, 7, 5)	(e_{10}, e_3, e_0), (e_8, e_5, e_0), (e_{13}, e_{10}, e_5),	(e_{11}, e_4, e_1), (e_9, e_6, e_1), (e_{14}, e_{11}, e_6),	(e_{12}, e_5, e_2), (e_{10}, e_7, e_2), (e_{12}, e_7, e_0),	(e_{13}, e_6, e_3), (e_{11}, e_8, e_3), (e_{13}, e_8, e_1),	(e_{14}, e_7, e_4), (e_{12}, e_9, e_4), (e_{14}, e_9, e_2),
(3, 8, 4)	(e_{11}, e_3, e_0), (e_8, e_5, e_1), (e_{13}, e_{10}, e_6),	(e_{12}, e_4, e_1), (e_9, e_6, e_2), (e_{14}, e_{11}, e_7),	(e_{13}, e_5, e_2), (e_{10}, e_7, e_3), (e_{12}, e_8, e_0),	(e_{14}, e_6, e_2), (e_{11}, e_8, e_4), (e_{13}, e_9, e_1),	(e_7, e_4, e_0), (e_{12}, e_9, e_5), (e_{14}, e_{10}, e_2),
(4, 4, 7)	(e_8, e_4, e_0), (e_{13}, e_9, e_5), (e_{14}, e_{10}, e_3),	(e_9, e_5, e_1), (e_{14}, e_{10}, e_6), (e_{11}, e_4, e_0),	(e_{10}, e_6, e_2), (e_{12}, e_8, e_0), (e_{12}, e_5, e_1),	(e_{11}, e_7, e_3), (e_{12}, e_8, e_1), (e_{13}, e_6, e_2),	(e_{12}, e_8, e_4), (e_{13}, e_9, e_2), (e_{14}, e_7, e_3),
(4, 5, 6)	(e_9, e_4, e_0), (e_{14}, e_9, e_5), (e_{14}, e_{10}, e_4),	(e_{10}, e_5, e_1), (e_{10}, e_6, e_0), (e_{11}, e_5, e_0),	(e_{11}, e_6, e_2), (e_{11}, e_7, e_1), (e_{12}, e_6, e_1),	(e_{12}, e_7, e_3), (e_{12}, e_8, e_2), (e_{13}, e_7, e_2),	(e_{13}, e_8, e_4), (e_{13}, e_9, e_3), (e_{14}, e_8, e_3),
(4, 6, 5)	(e_{10}, e_4, e_0), (e_9, e_5, e_0), (e_{14}, e_{10}, e_5),	(e_{11}, e_5, e_1), (e_{10}, e_6, e_1), (e_{11}, e_6, e_0),	(e_{12}, e_6, e_2), (e_{11}, e_7, e_2), (e_{12}, e_7, e_1),	(e_{13}, e_7, e_3), (e_{12}, e_8, e_3), (e_{13}, e_8, e_2),	(e_{14}, e_8, e_4), (e_{13}, e_9, e_4), (e_{14}, e_9, e_3),
(5, 5, 5)	(e_{10}, e_5, e_0),	(e_{11}, e_6, e_1),	(e_{12}, e_7, e_2),	(e_{13}, e_8, e_3),	(e_{14}, e_9, e_4),

З останнього рядка табл. 22 бачимо, що в моделі відстаней помилок (5, 5, 5) помилкові біти знаходяться на однаковій відстані один від одного. Якщо у такому разі виконувати порозрядний зсув бітів, то в кожному п'ятому такті помилкові біти накладатимуться один на одного (рис. 37). Інакше, в моделі відстаней помилок (5, 5, 5) помилки зустрічаються через кожні п'ять розрядів,

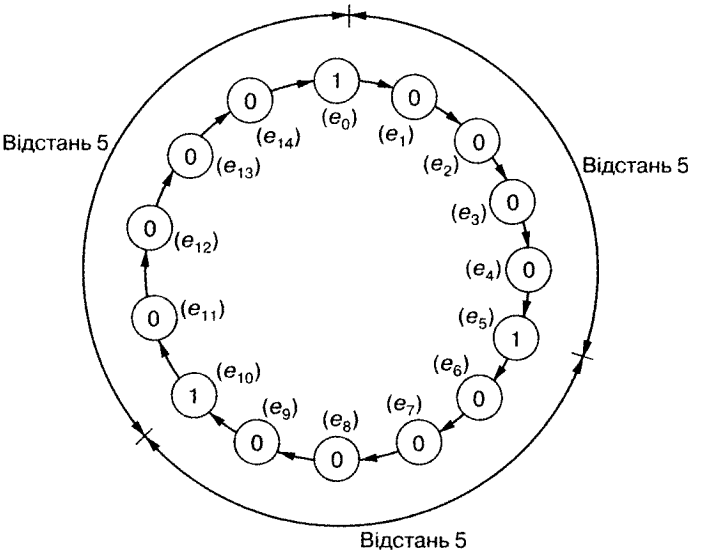


Рис. 37. Модель (5, 5, 5) відстаней між помилковими бітами

а це дозволяє виявити тільки п'ять кодових конфігурацій з помилками в трьох бітах.

У разі моделі відстаней помилок (5, 5, 5), що характеризується однаковими відстанями між помилковими бітами, проміжок часу, протягом якого виявляється помилкова конфігурація з помилками в трьох бітах, визначається так:

$$\frac{15(\textit{число розрядів коду})}{3(\textit{число помилкових бітів})} = 5\begin{pmatrix}\textit{число тактів, які потрібні для}\\ \textit{визначення помилкової конфігурації}\end{pmatrix}.$$

Отже, будь-яку кодову конфігурацію в моделі відстаней помилок (5, 5, 5) можна виявити протягом 1/3 періоду (період дорівнює 15 тактам), або за 5 тактів. Тоді після п'яти тактів вихід схеми визначення конфігурацій синдромів, необхідних для виявлення помилкових конфігурацій, повинен відключатися. Проте у разі використання моделі відстаней помилок (5, 5, 5) в процесі зсуву конфігурація синдрому визначається три рази, навіть якщо вихід схеми визначення конфігурацій синдромів після п'яти тактів не відключається:

виправлення помилки → виявлення помилки → виправлення помилки.

Зрештою помилкова конфігурація виправляється. Отже, у разі невеликого спрощення схеми кодові комбінації оброблятимуться так само, як і раніше.

Якщо код БЧХ містить n розрядів, а помилкові розряди знаходяться на однаковій відстані один від одного, то число помилкових розрядів обов'язково виражається через загальне число розрядів n. У такому разі можуть виявлятися помилкові кодові комбінації $\frac{n}{l}$ (l — число помилкових розрядів, що знаходяться на однаковій відстані один від одного).

Вихід схеми визначення конфігурацій синдромів повинен відключатися через $\frac{n}{l}$ тактів, проте у разі кодів БЧХ число розрядів завжди непарне, тому відключення схеми визначення конфігурацій синдромів після проходження $\frac{n}{l}$ тактів не є обов'язковим.

Як було показано вище, існує 31 група конфігурацій з помилками в трьох розрядах. З'ясуємо, чи всі помилкові конфігурації цих груп можна виявити.

Крім конфігурацій з рівною відстанню між помилковими бітами в табл. 22 наведено ще 30 груп конфігурацій з помилками в трьох розрядах. У кожній з цих груп можна виявити 15 конфігурацій, а в 31-й групі — п'ять конфігурацій з помилками в трьох розрядах. Отже, загальне число виявних конфігурацій з помилками в трьох розрядах становить

$$15 \cdot 30 + 5 \cdot 1 = 455,$$

а загальне число конфігурацій з помилками у трьох розрядах —

$$\binom{15}{3} = 455.$$

Таким чином, можна виявити всі помилкові конфігурації з помилками у трьох розрядах. Для визначення конфігурацій синдромів, необхідних для виявлення помилкових кодових конфігурацій, відразу ж виконується ділення кодів, що приймаються, на породжувальний многочлен.

Для моделі відстаней помилок (15) результат ділення коду, що приймається, з помилкою в одному розряді можна подати як $R(\alpha) = 1$, а результат ділення для моделі відстаней помилок з помилками в двох розрядах —

$$R(\alpha) = \alpha^j + 1. \tag{19}$$

Результат ділення коду, що приймається, для моделі відстаней помилок (i, j, k) з помилками в трьох розрядах можна записати так:

$$R(\alpha) = \alpha^{j+1} + \alpha^i + 1. \tag{20}$$

Оскільки код, що приймається, ділиться на породжувальний многочлен, степінь результату ділення менший, ніж степінь породжувального многочлена. У виразах (19) і (20) α — корінь породжувального многочлена $x^{10} + x^8 + x^5 + x^4 + x^2 + x^1 + 1$, тому

$$\alpha^{10} + \alpha^8 + \alpha^5 + \alpha^4 + \alpha^2 + \alpha^1 + 1 = 0,$$

тобто

$$\alpha^{10} = \alpha^8 + \alpha^5 + \alpha^4 + \alpha^2 + \alpha + 1. \tag{21}$$

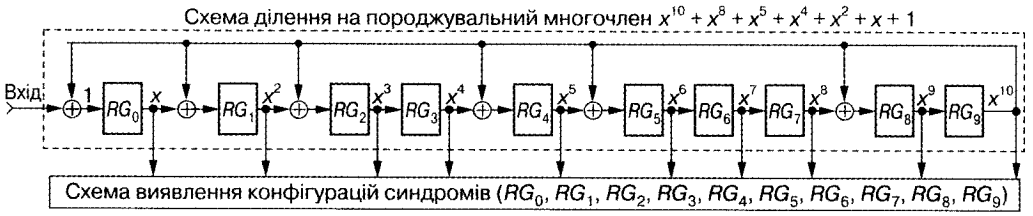
Оскільки степінь породжувального многочлена дорівнює 10, то породжувальні многочлени зі степенем вище ніж 10, згідно з формулою (21), можна виразити через α в степені не більше ніж 9. Тоді

$$\begin{aligned}
\alpha^{10} &= \alpha^8 + \alpha^5 + \alpha^4 + \alpha^2 + \alpha + 1,\\
\alpha^{11} &= \alpha \cdot \alpha^{10} = \alpha^9 + \alpha^6 + \alpha^5 + \alpha^3 + \alpha^2 + \alpha,\\
\alpha^{12} &= \alpha \cdot \alpha^{11} = \alpha^{10} + \alpha^7 + \alpha^6 + \alpha^4 + \alpha^3 + \alpha^2 =\\
&= \alpha^8 + \alpha^7 + \alpha^6 + \alpha^5 + \alpha^3 + \alpha + 1,\\
\alpha^{13} &= \alpha \cdot \alpha^{12} = \alpha^9 + \alpha^8 + \alpha^7 + \alpha^6 + \alpha^4 + \alpha^2 + \alpha,\\
\alpha^{14} &= \alpha \cdot \alpha^{13} = \alpha^{10} + \alpha^9 + \alpha^8 + \alpha^7 + \alpha^5 + \alpha^3 + \alpha^2 =\\
&= \alpha^9 + \alpha^7 + \alpha^4 + \alpha^3 + \alpha + 1.
\end{aligned} \tag{22}$$

Унаслідок ділення кодів (формули (19) та (20)), що приймаються, всі доданки з α в степені вище ніж 10 виражаються тільки через α в степені не більше 9. У такому разі відповідні конфігурації синдромів знаходяться досить просто. Моделі відстаней помилок і відповідні їм конфігурації синдромів наведені на рис. 38.

Схема декодування кодів БЧХ (15, 5), що виявляє потрійні помилки, показана на рис. 39.

Розглянемо приклад декодування коду з помилками в другому a_3, четвертому a_1 і восьмому розрядах c_7. Тоді помилкова кодова комбінація має вигляд (e_{13}, e_{11}, e_7), тобто, $e_{13} = 1$, $e_{11} = 1$, $e_7 = 1$, а решта розрядів дорівнює нулю. Помилкова кодова комбінація (e_{13}, e_{11}, e_7) належить моделі відстаней помилок (2, 9, 4), а синдром для її визначення має вигляд (1101011001) (див. рис. 38).



Модель відстаней помилки	Залишок від ділення на породжувальний многочлен	Конфігурація синдромів									
		RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7	RG_8	RG_9
(5)	$R(\alpha) = 1$	(1	0	0	0	0	0	0	0	0	0)
(1, 14)	$R(\alpha) = 1 + \alpha$	(1	1	0	0	0	0	0	0	0	0)
(2, 13)	$R(\alpha) = 1 + \alpha^2$	(1	0	1	0	0	0	0	0	0	0)
(3, 12)	$R(\alpha) = 1 + \alpha^3$	(1	0	0	1	0	0	0	0	0	0)
(4, 11)	$R(\alpha) = 1 + \alpha^4$	(1	0	0	0	1	0	0	0	0	0)
(5, 10)	$R(\alpha) = 1 + \alpha^5$	(1	0	0	0	0	1	0	0	0	0)
(6, 9)	$R(\alpha) = 1 + \alpha^6$	(1	0	0	0	0	0	1	0	0	0)
(7, 8)	$R(\alpha) = 1 + \alpha^7$	(1	0	0	0	0	0	0	1	0	0)
(1, 1, 13)	$R(\alpha) = 1 + \alpha + \alpha^2$	(1	1	1	0	0	0	0	0	0	0)
(1, 2, 12)	$R(\alpha) = 1 + \alpha + \alpha^3$	(1	1	0	1	0	0	0	0	0	0)
(1, 3, 11)	$R(\alpha) = 1 + \alpha + \alpha^4$	(1	1	0	0	1	0	0	0	0	0)
(1, 4, 10)	$R(\alpha) = 1 + \alpha + \alpha^5$	(1	1	0	0	0	1	0	0	0	0)
(1, 5, 9)	$R(\alpha) = 1 + \alpha + \alpha^6$	(1	1	0	0	0	0	1	0	0	0)
(1, 6, 8)	$R(\alpha) = 1 + \alpha + \alpha^7$	(1	1	0	0	0	0	0	1	0	0)
(1, 7, 7)	$R(\alpha) = 1 + \alpha + \alpha^8$	(1	1	0	0	0	0	0	0	1	0)
(1, 8, 6)	$R(\alpha) = 1 + \alpha + \alpha^9$	(1	1	0	0	0	0	0	0	0	1)
(1, 9, 5)	$R(\alpha) = 1 + \alpha + \alpha^{10} = \alpha^2 + \alpha^4 + \alpha^5 + \alpha^8$	(0	0	1	0	1	1	0	0	1	0)
(1, 10, 4)	$R(\alpha) = 1 + \alpha + \alpha^{11} = 1 + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^6 + \alpha^9$	(1	0	1	1	0	1	1	0	0	1)
(1, 11, 3)	$R(\alpha) = 1 + \alpha + \alpha^{12} = \alpha^3 + \alpha^5 + \alpha^6 + \alpha^7 + \alpha^8$	(0	0	0	1	0	1	1	1	1	0)
(1, 12, 2)	$R(\alpha) = 1 + \alpha + \alpha^{13} = 1 + \alpha^2 + \alpha^4 + \alpha^6 + \alpha^7 + \alpha^8 + \alpha^9$	(1	0	1	0	1	0	1	1	1	1)
(2, 2, 11)	$R(\alpha) = 1 + \alpha^2 + \alpha^4$	(1	0	1	0	1	0	0	0	0	0)
(2, 3, 10)	$R(\alpha) = 1 + \alpha^2 + \alpha^5$	(1	0	1	0	0	1	0	0	0	0)
(2, 4, 9)	$R(\alpha) = 1 + \alpha^2 + \alpha^6$	(1	0	1	0	0	0	1	0	0	0)
(2, 5, 8)	$R(\alpha) = 1 + \alpha^2 + \alpha^7$	(1	0	1	0	0	0	0	1	0	0)
(2, 6, 7)	$R(\alpha) = 1 + \alpha^2 + \alpha^8$	(1	0	1	0	0	0	0	0	1	0)
(2, 7, 6)	$R(\alpha) = 1 + \alpha^2 + \alpha^9$	(1	0	1	0	0	0	0	0	0	1)
(2, 8, 5)	$R(\alpha) = 1 + \alpha^2 + \alpha^{10} = \alpha + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^8$	(0	1	1	0	1	1	0	0	1	0)
(2, 9, 4)	$R(\alpha) = 1 + \alpha^2 + \alpha^{11} = 1 + \alpha + \alpha^3 + \alpha^5 + \alpha^6 + \alpha^9$	(1	1	0	1	0	1	1	0	0	1)
(2, 10, 3)	$R(\alpha) = 1 + \alpha^2 + \alpha^{12} = \alpha + \alpha^2 + \alpha^3 + \alpha^5 + \alpha^6 + \alpha^7 + \alpha^8$	(0	1	1	1	0	1	1	1	1	0)
(3, 3, 9)	$R(\alpha) = 1 + \alpha^2 + \alpha^6$	(1	0	0	1	0	0	1	0	0	0)
(3, 4, 8)	$R(\alpha) = 1 + \alpha^2 + \alpha^7$	(1	0	0	1	0	0	0	1	0	0)
(3, 5, 7)	$R(\alpha) = 1 + \alpha^2 + \alpha^8$	(1	0	0	1	0	0	0	0	1	0)
(3, 6, 6)	$R(\alpha) = 1 + \alpha^2 + \alpha^9$	(1	0	0	1	0	0	0	0	0	1)
(3, 7, 5)	$R(\alpha) = 1 + \alpha^3 + \alpha^{10} = \alpha + \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6$	(0	1	1	1	1	1	0	0	1	0)
(3, 8, 4)	$R(\alpha) = 1 + \alpha^3 + \alpha^{11} = 1 + \alpha + \alpha^2 + \alpha^5 + \alpha^6 + \alpha^9$	(1	1	1	0	0	1	1	0	0	1)
(4, 4, 7)	$R(\alpha) = 1 + \alpha^4 + \alpha^8$	(1	0	0	0	1	0	0	0	1	0)
(4, 5, 6)	$R(\alpha) = 1 + \alpha^4 + \alpha^9$	(1	0	0	0	1	0	0	0	0	1)
(4, 6, 5)	$R(\alpha) = 1 + \alpha^4 + \alpha^{10} = \alpha + \alpha^2 + \alpha^5 + \alpha^8$	(0	1	1	0	0	1	0	0	1	0)
(5, 5, 5)	$R(\alpha) = 1 + \alpha^5 + \alpha^{10} = \alpha + \alpha^2 + \alpha^4 + \alpha^8$	(0	1	1	0	1	0	0	0	1	0)

Рис. 38. Моделі відстаней помилок і відповідні їм конфігурації синдромів у випадку потрійних помилок

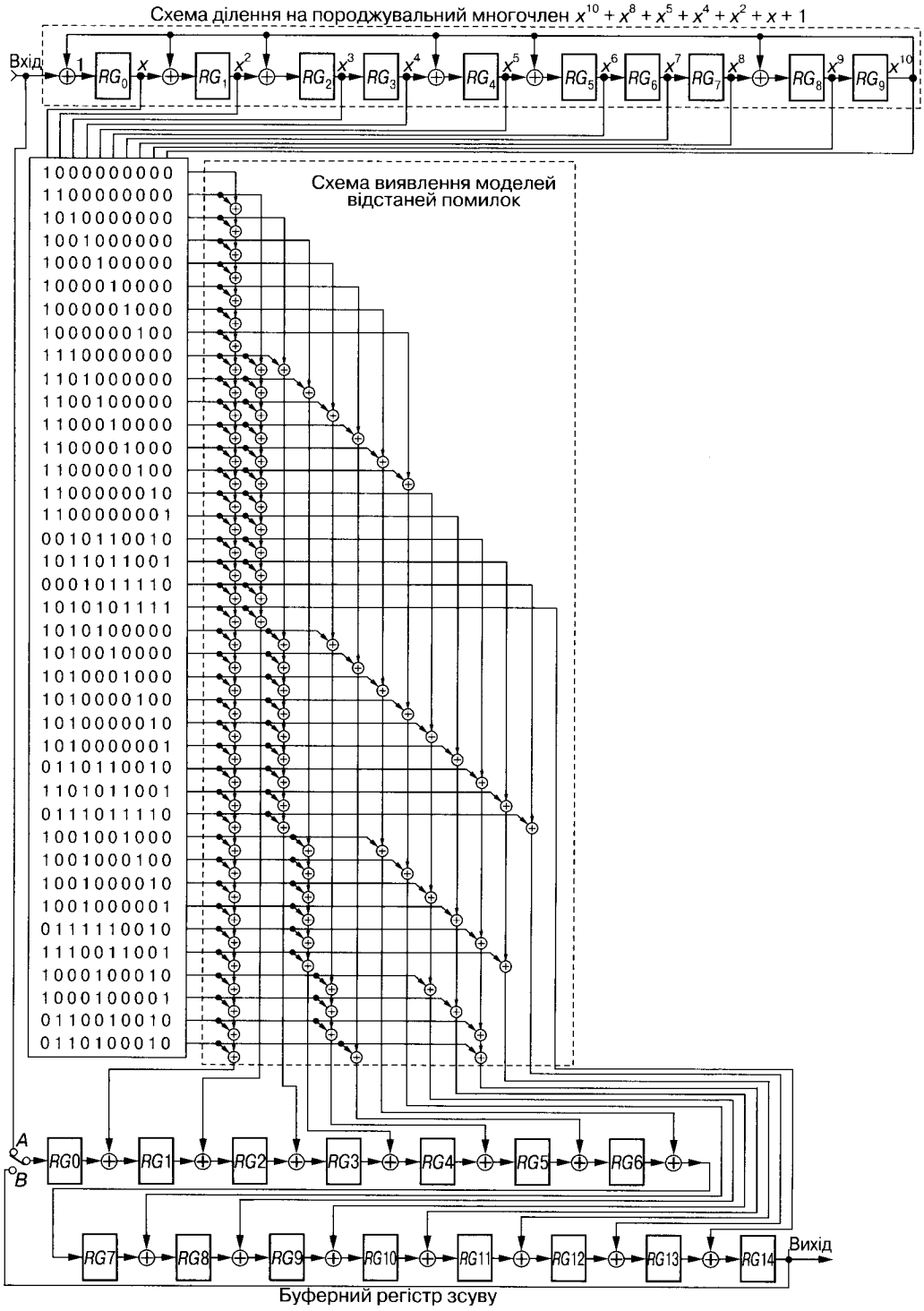


Рис. 39. Схема декодування кодів БЧХ (15, 5) для виявлення потрійних помилок

Таблиця 23. Приклад виправлення

Перемикач	Такт	Ділення на породжувальний многочлен									
		RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7	RG_8	RG_9
B	0	0	$e_{13} + e_{11}$	$e_{13} + e_{11}$	e_{11}	e_{13}	e_{11}	$e_{13} + e_{11}$	$e_{13} + e_7$	e_{13}	$e_{13} + e_{11}$
	1	$e_{13} + e_{11}$	$e_{13} + e_{11}$	0	$e_{13} + e_{11}$	e_{13}	e_{11}	e_{11}	$e_{13} + e_{11}$	$e_{11} + e_7$	e_{13}
	2	e_{13}	e_{11}	e_{11}	0	e_{11}	0	e_{11}	e_{11}	e_{11}	$e_{11} + e_7$
	3	$e_{11} + e_7$	$e_{13} + e_{11} + e_7$	e_7	e_{11}	$e_{11} + e_7$	e_7	0	e_{11}	e_7	e_{11}
	4	e_{11}	e_7	$e_{13} + e_7$	e_7	0	e_7	e_7	0	0	e_7
	5	e_7	$e_{11} + e_7$	0	$e_{13} + e_7$	0	e_7	e_7	e_7	e_7	0
	6	0	e_7	$e_{11} + e_7$	0	$e_{13} + e_7$	0	e_7	e_7	e_7	e_7
	7	e_7	e_7	0	$e_{11} + e_7$	e_7	e_{13}	0	e_7	0	e_7
	8	e_7	0	0	0	e_{11}	0	e_{13}	0	0	0
	9	0	e_7	0	0	0	e_{11}	0	e_{13}	0	0
	10	0	0	e_7	0	0	0	e_{11}	0	e_{13}	0
	11	0	0	0	e_7	0	0	0	e_{11}	0	e_{13}
	12	e_{13}	e_{13}	e_{13}	0	$e_{13} + e_7$	e_{13}	0	0	$e_{13} + e_{11}$	0
	13	0	e_{13}	e_{13}	e_{13}	0	$e_{13} + e_7$	e_{13}	0	0	$e_{13} + e_{11}$
	14	$e_{13} + e_{11}$	$e_{13} + e_{11}$	e_{11}	e_{13}	e_{11}	$e_{13} + e_{11}$	$e_{13} + e_{11}$	e_{13}	$e_{13} + e_{11}$	0
A	15	—	—	—	—	—	—	—	—	—	—
	16	—	—	—	—	—	—	—	—	—	—
	17	—	—	—	—	—	—	—	—	—	—
	18	—	—	—	—	—	—	—	—	—	—
	19	—	—	—	—	—	—	—	—	—	—
	20	—	—	—	—	—	—	—	—	—	—
	21	—	—	—	—	—	—	—	—	—	—
	22	—	—	—	—	—	—	—	—	—	—
	23	—	—	—	—	—	—	—	—	—	—
	24	—	—	—	—	—	—	—	—	—	—
	25	—	—	—	—	—	—	—	—	—	—
	26	—	—	—	—	—	—	—	—	—	—
	27	—	—	—	—	—	—	—	—	—	—
	28	—	—	—	—	—	—	—	—	—	—
	29	—	—	—	—	—	—	—	—	—	—

Процес виправлення цієї потрійної помилки продемонстрований за допомогою табл. 23.

Як бачимо з табл. 23, виправлення помилок виконується відразу після прийому всіх бітів, що відповідають розрядам коду. У момент закінчення прийому прийнята послідовність ділиться на породжувальний многочлен:

$$R(\alpha) = e_{14}\alpha^{14} + (e_{13})\alpha^{13} + e_{12}\alpha^{12} + (e_{11})\alpha^{11} + e_{10}\alpha^{14} + e_{10}\alpha^{10} + e_9\alpha^9 + \\ + e_8\alpha^8 + (e_7)\alpha^7 + e_6\alpha^6 + e_5\alpha^5 + e_4\alpha^4 + e_3\alpha^3 + e_2\alpha^2 + e_1\alpha + e_0. \qquad (23)$$

потрійних помилок

Вхід схеми виявлення конфігурацій синдромів	Приймальний буферний регістр															Вихід
	*RG*0	*RG*1	*RG*2	*RG*3	*RG*4	*RG*5	*RG*6	*RG*7	*RG*8	*RG*9	*RG*10	*RG*11	*RG*12	*RG*13	*RG*14	
(0001110010)	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\bar{c}_7$	c_8	c_9	a_0	$\bar{a}_1$	a_2	a_3	a_4	—
(0000111001)	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\bar{c}_7$	c_8	c_9	a_0	$\bar{a}_1$	a_2	a_3	—
(1110101110)	$\bar{a}_3$	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\bar{c}_7$	c_8	c_9	a_0	$\bar{a}_1$	a_2	—
(0111010111)	a_2	$\bar{a}_3$	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\bar{c}_7$	c_8	c_9	a_0	$\bar{a}_1$	—
(0101011010)	$\bar{a}_1$	a_2	$\bar{a}_3$	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	$\bar{c}_7$	c_8	c_9	a_0	—
Конфігурація синдрому для моделі відстаней помилок (2, 9, 4 → Вихід виявлення "1"	→ Вихід схеми виявлення моделей відстані помилок															
	1	(0)	1	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	1	(0)	(0)	(0)	
(1000011110)	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	—
(0100001111)	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	—
(0100110101)	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	—
(1000101000)	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	c_4	c_5	c_6	—
(0100010100)	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	c_4	c_5	—
(0010001010)	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	c_4	—
(0001000101)	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	c_3	—
(1110010010)	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	c_2	—
(0111001001)	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	c_1	—
(0011100100)	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	c_0	—
—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	a_4
—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_3
—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_2
—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_1	a_1
—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	a_0	a_0
—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_9	c_9
—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_8	c_8
—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_7	c_7
—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_6	c_6
—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_5	c_5
—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_4	c_4
—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_3	c_3
—	—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_2	c_2
—	—	—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_1	c_1
—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	c_0	c_0

Лише e_3, e_{11}, $e_7 = 1$, інші дорівнюють "0"
$\bar{a}_3 = a_3 + e_{13}$, $\bar{a}_1 = a_1 + e_{11}$, $\bar{c}_7 = c_7 + e_7$

де $e_{13} = e_{11} = e_7 = 1$, а в решті розрядів записані нулі.

Степінь результату від ділення має бути меншою, ніж степінь породжувального многочлена, тому, використовуючи формули (22), можна доданки з α в степені більше ніж 10 замінити доданками з α меншими або зрівнянними 9. При цьому одержуємо

$$R(\alpha) = (e_{14} + \langle e_{13}\rangle + \langle e_{11}\rangle + e_9)\alpha^9 + (\langle e_{13}\rangle + e_{12} + e_{10} + e_8)\alpha^8 +$$

$$+ (e_{14} + \langle e_{13}\rangle + e_{12} + \langle e_7\rangle)\alpha^7 + (\langle e_{13}\rangle + e_{12} + \langle e_{11}\rangle + e_6)\alpha^6 +$$
$$+ (e_{12} + \langle e_{11}\rangle + e_{10} + e_5)\alpha^5 + (e_{14} + \langle e_{13}\rangle + e_{10} + e_4)\alpha^4 +$$
$$+ (e_{14} + e_{12} + \langle e_{11}\rangle + e_3)\alpha^3 + (\langle e_{13}\rangle + \langle e_{11}\rangle + e_{10} + e_2)\alpha^2 +$$
$$+ (e_{14} + \langle e_{13}\rangle + e_{12} + \langle e_{11}\rangle + e_{10} + e_1)\alpha + (e_{14} + e_{12} + e_{10} + e_0)\alpha^9. \qquad (24)$$

Таким чином,

$$R(\alpha) = (\langle e_{13}\rangle + \langle e_{11}\rangle)\alpha^9 + \langle e_{13}\rangle\alpha^8 + (\langle e_{13}\rangle + \langle e_7\rangle)\alpha^7 + (\langle e_{13}\rangle + \langle e_{11}\rangle)\alpha^6 +$$
$$+ \langle e_{11}\rangle\alpha^5 + \langle e_{13}\rangle\alpha^4 + \langle e_{11}\rangle\alpha^3 + (\langle e_{13}\rangle + \langle e_{11}\rangle)\alpha^2 + (\langle e_{13}\rangle + \langle e_{11}\rangle)\alpha.$$

Цей результат від ділення після закінчення прийому бітів відповідних розрядів кодової послідовності, що приймається, відображається розрядами регістра зсуву схеми ділення на породжувальний многочлен.

18.1.4. Код БЧХ (31, 21) DEC

Тут у деякій мірі повторюється викладене вище. Проте для вільного оперування з кодами БЧХ необхідно неодноразово виконати процедури формування (кодування) і декодування цих кодів. Тому нижче наведено узагальнені принципи декодування та виправлення помилок у кодах БЧХ.

Раніше для формування коду БЧХ вибирався вихідний (початковий) многочлен четвертого степеня, а число розрядів коду становило: $n = 2^4 - 1 = 15$. Тут розглядається код БЧХ з виправленням помилок у двох розрядах (DEC), для побудови якого використовується початковий многочлен п'ятого степеня $x^5 + x^2 + 1$, число розрядів коду якого визначається так: $n = 2^5 - 1 = 31$.

Проаналізуємо, як декодуються кодові комбінації цього коду.

Якщо корінь вихідного многочлена $x^5 + x^2 + 1$ позначити через α, то код формуватиметься з породжувального многочлена, коренями якого є результати піднесення α до першого—четвертого степеня, тобто послідовність α, α^2, α^3, α^4.

Для отримання породжувального многочлена необхідно визначити найменший многочлен з коренями α, α^2, α^3, α^4. Оскільки α є коренем початкового многочлена $x^5 + x^2 + 1$, то він є і початковим елементом скінченного поля Галуа $GF(2^5)$, що утворюється унаслідок ділення цього вихідного многочлена. Тому знаходячи найменший многочлен з коренями α, α^2, α^3, α^4, елементи $GF(2^5)$:

$$\alpha, \alpha^2, \alpha^3, \alpha^4, ..., \alpha^{30}, \alpha^{31} (= \alpha^0)$$

(за винятком нульового) зручно виражати через початковий многочлен з коренем α. Оскільки α — корінь початкового многочлена $x^5 + x^2 + 1$:

$$\alpha^5 + \alpha^2 + 1 = 0,$$

то, використовуючи залежність

$$\alpha^5 = \alpha^2 + 1,$$

усі елементи поля Галуа $GF(2^5)$ (за винятком нульового) можна виразити через многочлени зі степенем α не вище ніж чотири, а саме:

$$
\begin{aligned}
\alpha^0 &= 1\\
\alpha &= \alpha\\
\alpha^2 &= \alpha^2\\
\alpha^3 &= \alpha^3\\
\alpha^4 &= \alpha^4\\
\alpha^5 &= \alpha^2 + 1\\
\alpha^6 &= \alpha^3 + \alpha\\
\alpha^7 &= \alpha^4 + \alpha^2\\
\alpha^8 &= \alpha^3 + \alpha^2 + 1\\
\alpha^9 &= \alpha^4 + \alpha^3 + \alpha\\
\alpha^{10} &= \alpha^4 + 1\\
\alpha^{11} &= \alpha^2 + \alpha + 1\\
\alpha^{12} &= \alpha^3 + \alpha^2 + \alpha\\
\alpha^{13} &= \alpha^4 + \alpha^3 + \alpha^2\\
\alpha^{14} &= \alpha^4 + \alpha^3 + \alpha^2 + 1\\
\alpha^{15} &= \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1\\
\alpha^{16} &= \alpha^4 + \alpha^3 + \alpha + 1\\
\alpha^{17} &= \alpha^4 + \alpha + 1\\
\alpha^{18} &= \alpha + 1\\
\alpha^{19} &= \alpha^2 + \alpha\\
\alpha^{20} &= \alpha^3 + \alpha^2\\
\alpha^{21} &= \alpha^4 + \alpha^3\\
\alpha^{22} &= \alpha^4 + \alpha^2 + 1\\
\alpha^{23} &= \alpha^3 + \alpha^2 + \alpha + 1\\
\alpha^{24} &= \alpha^4 + \alpha^3 + \alpha^2 + \alpha\\
\alpha^{25} &= \alpha^4 + \alpha^3 + 1\\
\alpha^{26} &= \alpha^4 + \alpha^2 + \alpha + 1\\
\alpha^{27} &= \alpha^3 + \alpha + 1\\
\alpha^{28} &= \alpha^4 + \alpha^2 + \alpha\\
\alpha^{29} &= \alpha^3 + 1\\
\alpha^{30} &= \alpha^4 + \alpha\\
\alpha^{31} &= 1\\
\alpha^{32} &= \alpha^0
\end{aligned}
\qquad (25)
$$

Спробуємо визначити найменший многочлен з коренями α, α^2, α^3, α^4, який є результатом піднесення до цілочислового степеня кореня α вихідного многочлена.

Відразу ж знайдемо найменший многочлен з коренем α. Оскільки ряд степенів α має вигляд

$$\alpha, \alpha^2, \alpha^4, \alpha^8, \alpha^{16}, \alpha^{32} = \alpha^{31} \cdot \alpha = \alpha, \alpha^2, \ldots,$$

то найменший многочлен теж знаходиться в ряду α, α^2, α^4, α^8, α^{16}. Коренями найменшого многочлена є α, α^2, α^4, α^8, α^{16}, тому найменший многочлен, який шукається, запишеться наступним співвідношенням:

$$(x-\alpha)(x-\alpha^2)(x-\alpha^4)(x-\alpha^8)(x-\alpha^{16}),$$

тобто

$$(x+\alpha)(x+\alpha^2)(x+\alpha^4)(x+\alpha^8)(x+\alpha^{16}) = x^5 + Ax^4 + Bx^3 + Cx^2 + Dx + E. \qquad (26)$$

Визначимо всі коефіцієнти A, B, C, D і E, що входять до цієї формули:
коефіцієнт A:

$$A = \alpha + \alpha^2 + \alpha^4 + \alpha^8 + \alpha^{16} = \alpha + \alpha^2 + \alpha^4 + (\alpha^3 + \alpha^2 + 1) + (\alpha^4 + \alpha^3 + \alpha + 1) = 0;$$

коефіцієнт B:

$$\begin{aligned} B &= \alpha\cdot\alpha^2 + \alpha\cdot\alpha^4 + \alpha\cdot\alpha^8 + \alpha\cdot\alpha^{16} + \alpha^2\cdot\alpha^4 + \\ &+\alpha^2\cdot\alpha^8 + \alpha^2\cdot\alpha^{16} + \alpha^4\cdot\alpha^8 + \alpha^4\cdot\alpha^{16} + \alpha^8\cdot\alpha^{16} = \\ &= \alpha^3 + \alpha^5 + \alpha^9 + \alpha^{17} + \alpha^6 + \alpha^{10} + \alpha^{18} + \alpha^{12} + \alpha^{20} + \alpha^{24} = \\ &= \alpha^3 + (\alpha^2+1) + (\alpha^4 + \alpha^3 + \alpha) + (\alpha^4 + \alpha + 1) + (\alpha^3 + \alpha) + (\alpha^4 + 1) + \\ &+(\alpha+1) + (\alpha^3 + \alpha^2 + \alpha) + (\alpha^3 + \alpha^2) + (\alpha^4 + \alpha^3 + \alpha^2 + \alpha) = 0; \end{aligned}$$

коефіцієнт C:

$$\begin{aligned} C &= \alpha\cdot\alpha^2\cdot\alpha^4 + \alpha\cdot\alpha^2\cdot\alpha^8 + \alpha\cdot\alpha^2\cdot\alpha^{16} + \\ &+\alpha\cdot\alpha^4\cdot\alpha^8 + \alpha\cdot\alpha^{14}\cdot\alpha^{16} + \alpha\cdot\alpha^8\cdot\alpha^{16} + \\ &+\alpha^2\cdot\alpha^4\cdot\alpha^8 + \alpha^2\cdot\alpha^4\cdot\alpha^{16} + \alpha^2\cdot\alpha^8\cdot\alpha^{16} + \alpha^4\cdot\alpha^8\cdot\alpha^{16} = \\ &= \alpha^7 + \alpha^{11} + \alpha^{19} + \alpha^{13} + \alpha^{21} + \alpha^{25} + \alpha^{14} + \alpha^{22} + \alpha^{26} + \alpha^{28} = \\ &= (\alpha^4 + \alpha^2) + (\alpha^2 + \alpha + 1) + (\alpha^2 + \alpha) + (\alpha^4 + \alpha^3 + \alpha^2) + \\ &+(\alpha^4 + \alpha^3) + (\alpha^4 + \alpha^3 + 1) + (\alpha^4 + \alpha^3 + \alpha^2 + 1) + \\ &+(\alpha^4 + \alpha^2 + 1) + (\alpha^4 + \alpha^2 + \alpha + 1) + (\alpha^4 + \alpha^2 + \alpha) = 1; \end{aligned}$$

коефіцієнт D:

$$\begin{aligned} D &= \alpha\cdot\alpha^2\cdot\alpha^4\cdot\alpha^8 + \alpha\cdot\alpha^2\cdot\alpha^4\cdot\alpha^{16} + \\ &+\alpha\cdot\alpha^2\cdot\alpha^8\cdot\alpha^{16} + \alpha\cdot\alpha^4\cdot\alpha^8\cdot\alpha^{16} + \alpha^2\cdot\alpha^4\cdot\alpha^8\cdot\alpha^{16} = \\ &= \alpha^{15} + \alpha^{23} + \alpha^{27} + \alpha^{29} + \alpha^{30} = \\ &= (\alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1) + \\ &+(\alpha^3 + \alpha^2 + \alpha + 1) + (\alpha^3 + \alpha + 1) + \\ &+(\alpha^3 + 1) + (\alpha^4 + \alpha) = 0; \end{aligned}$$

коефіцієнт E:

$$E = \alpha\cdot\alpha^2\cdot\alpha^4\cdot\alpha^8\cdot\alpha^{16} = \alpha^{31} = 1.$$

Отже, найменшим многочленом з коренем α є многочлен $x^5 + x^2 + 1$. Тобто, оскільки α є коренем вихідного многочлена, найменшим многочленом з коренем α є також вираз $x^5 + x^2 + 1$ необхідності його повторного визначення немає.

Найменший многочлен з коренем α є і найменшим многочленом з коренями α^2, α^4, α^8, α^{16}. Отже, три корені α, α^2, α^4 з чотирьох: α, α^2, α^3, α^4 — корені найменшого многочлена $x^5 + x^2 + 1$.

Залишається визначити лише найменший многочлен для кореня α^3. У цьому випадку ряд має вигляд

$$\alpha^3, \alpha^6, \alpha^{12}, \alpha^{24}, \alpha^{48} = \alpha^{31} \cdot \alpha^{17} = \alpha^{17}, \alpha^{34} = \alpha^{31} \cdot \alpha^3 = \alpha^3, \alpha^6, \ldots,$$

тому найменший многочлен теж знаходиться в ряду α^3, α^6, α^{12}, α^{24}, α^{17}. Отже, найменший многочлен з коренем α^3 запишеться так:

$$(x - \alpha^3)(x - \alpha^6)(x - \alpha^{12})(x - \alpha^{24})(x - \alpha^{17}),$$

тобто

$$(x + \alpha^3)(x + \alpha^6)(x + \alpha^{12})(x + \alpha^{24})(x + \alpha^{17}) = \\ = x^5 + Ax^4 + Bx^3 + Cx^2 + Dx + E. \qquad (27)$$

Визначаючи коефіцієнти A, B, C, D і E, що входять у формулу (27), отримуємо:

коефіцієнт A:

$$A = \alpha^3 + \alpha^6 + \alpha^{12} + \alpha^{24} + \alpha^{17} = \\ = \alpha^3 + (\alpha^3 + \alpha) + (\alpha^3 + \alpha^2 + \alpha) + \\ + (\alpha^4 + \alpha^3 + \alpha^2 + \alpha) + (\alpha^4 + \alpha + 1) = 1;$$

коефіцієнт B:

$$B = \alpha^3 \cdot \alpha^6 + \alpha^3 \cdot \alpha^{12} + \alpha^3 \cdot \alpha^{24} + \alpha^3 \cdot \alpha^{17} + \alpha^6 \cdot \alpha^{12} + \alpha^6 \cdot \alpha^{24} + \alpha^6 \cdot \alpha^{17} + \\ + \alpha^{12} \cdot \alpha^{24} + \alpha^{12} \cdot \alpha^{17} + \alpha^{24} \cdot \alpha^{17} = \alpha^9 + \alpha^{15} + \alpha^{27} + \alpha^{20} + \alpha^{18} + \alpha^{30} + \\ + \alpha^{23} + \alpha^{36} + \alpha^{29} + \alpha^{41} = (\alpha^4 + \alpha^3 + 1) + (\alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1) + \\ + (\alpha^3 + \alpha + 1) + (\alpha^3 + \alpha^2) + (\alpha + 1) + (\alpha^4 + \alpha) + (\alpha^3 + \alpha^2 + \alpha + 1) + \\ + (\alpha^2 + 1) + (\alpha^3 + 1) + (\alpha^4 + 1) = 1;$$

коефіцієнт C:

$$C = \alpha^3 \cdot \alpha^6 \cdot \alpha^{12} + \alpha^3 \cdot \alpha^6 \cdot \alpha^{24} + \alpha^3 \cdot \alpha^6 \cdot \alpha^{17} + \alpha^3 \cdot \alpha^{12} \cdot \alpha^{24} + \\ + \alpha^3 \cdot \alpha^{12} \cdot \alpha^{17} + \alpha^3 \cdot \alpha^{24} \cdot \alpha^{17} + \alpha^6 \cdot \alpha^{12} \cdot \alpha^{24} + \alpha^6 \cdot \alpha^{12} \cdot \alpha^{17} + \\ + \alpha^6 \cdot \alpha^{24} \cdot \alpha^{17} + \alpha^{12} \cdot \alpha^{24} \cdot \alpha^{17} = \\ = \alpha^{21} + \alpha^{33} + \alpha^{26} + \alpha^{39} + \alpha^{32} + \alpha^{44} + \alpha^{42} + \alpha^{35} + \alpha^{47} + \alpha^{53} = \\ = (\alpha^4 + \alpha^3) + \alpha^2 + (\alpha^4 + \alpha^2 + \alpha + 1) + \\ + (\alpha^3 + \alpha^2 + 1) + \alpha + (\alpha^4 + \alpha^3 + \alpha^2) + \\ + (\alpha^2 + \alpha + 1) + \alpha^4 + (\alpha^4 + \alpha^3 + \alpha + 1) + (\alpha^4 + \alpha^2 + 1) = 1;$$

коефіцієнт D:

$$D = \alpha^3 \cdot \alpha^6 \cdot \alpha^{12} \cdot \alpha^{24} + \alpha^3 \cdot \alpha^6 \cdot \alpha^{12} \cdot \alpha^{17} +$$
$$+\alpha^3 \cdot \alpha^6 \cdot \alpha^{24} \cdot \alpha^{17} + \alpha^3 \cdot \alpha^{12} \cdot \alpha^{24} \cdot \alpha^{17} + \alpha^6 \cdot \alpha^{12} \cdot \alpha^{24} \cdot \alpha^{17} =$$
$$= \alpha^{45} + \alpha^{38} + \alpha^{50} + \alpha^{56} + \alpha^{59} =$$
$$= (\alpha^4 + \alpha^3 + \alpha^2 + 1) + (\alpha^4 + \alpha^2) + (\alpha^2 + \alpha) +$$
$$+(\alpha^4 + \alpha^3 + 1) + (\alpha^4 + \alpha^2 + \alpha) = 0;$$

коефіцієнт E:

$$E = \alpha^3 \cdot \alpha^6 \cdot \alpha^{12} \cdot \alpha^{24} \cdot \alpha^{17} = \alpha^{62} = 1.$$

Отже, найменшим многочленом з коренем α^3 є многочлен $x^5 + x^4 + x^3 + x^2 + 1$.

Отримані результати дають змогу визначити найменший многочлен з коренями α, α^2, α^3, α^4, що є результатом піднесення α до першого—четвертого степеня. Іншими словами, найменший многочлен з коренями α, α^2, α^4 — це $x^5 + x^2 + 1$; найменший многочлен з коренем α^3 — це $x^5 + x^4 + x^3 + x^2 + 1$. Таким чином, породжувальний многочлен, коренями якого є результати піднесення α до першого — четвертого степеня, тобто α, α^2, α^3, α^4, має наступний вигляд:

Породжувальний многочлен = LCM {*найменший многочлен з коренем* α,
найменший многочлен з коренем α^2,
найменший многочлен з коренем α^3,
найменший многочлен з коренем α^4} =
$= \text{LCM}\{x^5 + x^2 + 1, x^5 + x^2 + 1, x^5 + x^4 + x^3 + x^2 + 1, x^5 + x^2 + 1\} =$
$= (x^5 + x^2 + 1)(x^5 + x^4 + x^3 + x^2 + 1) =$
$= x^{10} + x^9 + x^8 + x^6 + x^5 + x^3 + 1.$

Оскільки максимальний степінь породжувального многочлена дорівнює 10, то число контрольних розрядів коду теж — 10.

З викладеного вище випливає наступна структура коду:

- число розрядів коду $n = 31$;
- число контрольних розрядів $m = 10$;
- число інформаційних розрядів $k = 21$;
- можливість виправлення помилок — DEC.

Як наслідок, з породжувального многочлена $x^{10} + x^9 + x^8 + x^6 + x^5 + x^3 + 1$ формується код БЧХ (31, 21), який виправляє подвійні помилки.

На рис. 36 показано приклад формування коду БЧХ (15, 5). Код БЧХ (31, 21) формується аналогічно. Число розрядів коду БЧХ (15, 5) дорівнює 15, отже, можна використовувати многочлен, максимальний степінь x якого дорівнює 14. У коді БЧХ (31, 21) число розрядів — 31, тому максимальний степінь x використовуваного многочлена — 30.

Оскільки число інформаційних розрядів — 21, інформаційна послідовність має вигляд $a = (a_{20}, a_{19}, a_{18}, ..., a_1, a_0)$, а інформаційний многочлен —

$$A(x) = a_{20}x^{20} + a_{19}x^{19} + a_{18}x^{18} + ... + a_1x + a_0.$$

Максимальний степінь породжувального многочлена дорівнює 10, тому 10 контрольних розрядів визначаються унаслідок ділення $A(x) \cdot x^{10}$ на породжувальний многочлен. У разі прямого ділення, що аналогічне наведеному на рис. 36, степінь многочлена становитиме 30, і він потребуватиме великих обсягів обчислень, тому спробуємо визначити залишок від ділення добутку $A(x) \cdot x^{10}$ за залишками від ділення кожного з доданків з x в степені більшому, ніж степінь породжувального многочлена.

Максимальний степінь породжувального многочлена

$$G(x) = x^{10} + x^9 + x^8 + x^6 + x^5 + x^3 + 1 \qquad (28)$$

дорівнює 10, тому визначатимемо залишок від ділення доданків з x в степені від 10 до 30. При цьому для окремих доданків отримаємо такі співвідношення:

$$\begin{aligned}
x^{10} &= x^9 + x^8 + x^6 + x^5 + x^3 + 1 && (\text{mod } G(x)),\\
x^{11} &= x \cdot x^{10} = x^8 + x^7 + x^5 + x^4 + x^3 + x + 1 && (\text{mod } G(x)),\\
x^{12} &= x \cdot x^{11} = x^9 + x^8 + x^6 + x^5 + x^4 + x^2 + x && (\text{mod } G(x)),\\
x^{13} &= x \cdot x^{12} = x^8 + x^7 + x^2 + 1 && (\text{mod } G(x)),\\
x^{14} &= x \cdot x^{13} = x^9 + x^8 + x^3 + x && (\text{mod } G(x)),\\
x^{15} &= x \cdot x^{14} = x^8 + x^6 + x^5 + x^4 + x^3 + x^2 + 1 && (\text{mod } G(x)),\\
x^{16} &= x \cdot x^{15} = x^9 + x^7 + x^6 + x^5 + x^4 + x^3 + x && (\text{mod } G(x)),\\
x^{17} &= x \cdot x^{16} = x^9 + x^7 + x^4 + x^3 + x^2 + 1 && (\text{mod } G(x)),\\
x^{18} &= x \cdot x^{17} = x^9 + x^6 + x^4 + x + 1 && (\text{mod } G(x)),\\
x^{19} &= x \cdot x^{18} = x^9 + x^8 + x^7 + x^6 + x^3 + x^2 + x + 1 && (\text{mod } G(x)),\\
x^{20} &= x \cdot x^{19} = x^7 + x^6 + x^5 + x^4 + x^2 + x + 1 && (\text{mod } G(x)),\\
x^{21} &= x \cdot x^{20} = x^8 + x^7 + x^6 + x^5 + x^3 + x^2 + x && (\text{mod } G(x)),\\
x^{22} &= x \cdot x^{21} = x^9 + x^8 + x^7 + x^6 + x^4 + x^3 + x^2 && (\text{mod } G(x)),\\
x^{23} &= x \cdot x^{22} = x^7 + x^6 + x^4 + 1 && (\text{mod } G(x)),\\
x^{24} &= x \cdot x^{23} = x^8 + x^7 + x^5 + x && (\text{mod } G(x)),\\
x^{25} &= x \cdot x^{24} = x^9 + x^8 + x^6 + x^2 && (\text{mod } G(x)),\\
x^{26} &= x \cdot x^{25} = x^8 + x^7 + x^6 + x^5 + 1 && (\text{mod } G(x)),\\
x^{27} &= x \cdot x^{26} = x^9 + x^8 + x^7 + x^4 + x && (\text{mod } G(x)),\\
x^{28} &= x \cdot x^{27} = x^7 + x^6 + x^5 + x^3 + x^2 + 1 && (\text{mod } G(x)),\\
x^{29} &= x \cdot x^{28} = x^8 + x^7 + x^6 + x^4 + x^3 + x && (\text{mod } G(x)),\\
x^{30} &= x \cdot x^{29} = x^9 + x^8 + x^7 + x^5 + x^4 + x^2 && (\text{mod } G(x)).
\end{aligned}$$

Отже, залишки від ділення доданків виразу

$$A(x)\cdot x^{10} = a_{20}x^{30} + a_{19}x^{29} + a_{18}x^{28} + \ldots + a_1x^{11} + a_0x^{10}$$

на породжувальний многочлен можна подати, як на рис. 40.

З рис. 40 видно, що залишок $R(x)$, отриманий при діленні $A(x)\cdot x^{10}$ на $G(x)$ виражається формулою

$$R(x) = c_9x^9 + c_8x^8 + c_7x^7 + c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0, \qquad (29)$$

де

$$c_9 = a_{20} + a_{17} + a_{15} + a_{12} + a_9 + a_8 + a_7 + a_6 + a_4 + a_2 \pmod 2;$$

$$c_8 = a_{20} + a_{19} + a_{17} + a_{16} + a_{15} + a_{14} + a_{12} + a_{11} + a_9 + a_5 + a_4 + a_3 + a_2 + a_1 + a_0 \pmod 2;$$

$$c_7 = a_{20} + a_{19} + a_{18} + a_{17} + a_{16} + a_{14} + a_{13} + a_{12} + a_{11} + a_{10} + a_9 + a_7 + a_6 + a_3 + a_1 \pmod 2;$$

$$c_6 = a_{19} + a_{18} + a_{16} + a_{15} + a_{13} + a_{12} + a_{11} + a_{10} + a_9 + a_8 + a_6 + a_5 + a_2 + a_0 \pmod 2;$$

Складові формули $A(x)\cdot x^{10}$	Залишок від ділення кожного доданку $A(x)\cdot x^{10}$ на $G(x)$									
	x^9	x^8	x^7	x^6	x^5	x^4	x^3	x^2	x^1	x^0
$a_{20}x^{30}$	$a_{20}x^9$	$+a_{20}x^8$	$+a_{20}x^7$		$+a_{20}x^5$	$+a_{20}x^4$		$+a_{20}x^2$		
$a_{19}x^{29}$		$+a_{19}x^8$	$+a_{19}x^7$	$+a_{19}x^6$		$+a_{19}x^4$	$+a_{19}x^3$		$+a_{19}x$	
$a_{18}x^{28}$			$+a_{18}x^7$	$+a_{18}x^6$	$+a_{18}x^5$		$+a_{18}x^3$	$+a_{18}x^2$		$+a_{18}$
$a_{17}x^{27}$	$a_{17}x^9$	$+a_{17}x^8$	$+a_{17}x^7$			$+a_{17}x^4$			$+a_{17}x$	
$a_{16}x^{26}$		$+a_{16}x^8$	$+a_{16}x^7$	$+a_{16}x^6$	$+a_{16}x^5$					$+a_{16}$
$a_{15}x^{25}$	$a_{15}x^9$	$+a_{15}x^8$		$+a_{15}x^6$				$+a_{15}x^2$		
$a_{14}x^{24}$		$+a_{14}x^8$	$+a_{14}x^7$		$+a_{14}x^5$				$+a_{14}x$	
$a_{13}x^{23}$			$+a_{13}x^7$	$+a_{13}x^6$		$+a_{13}x^4$				$+a_{13}$
$a_{12}x^{22}$	$a_{12}x^9$	$+a_{12}x^8$	$+a_{12}x^7$	$+a_{12}x^6$		$+a_{12}x^4$	$+a_{12}x^3$	$+a_{12}x^2$		
$a_{11}x^{21}$		$+a_{11}x^8$	$+a_{11}x^7$	$+a_{11}x^6$	$+a_{11}x^5$		$+a_{11}x^3$	$+a_{11}x^2$	$+a_{11}x$	
$a_{10}x^{20}$			$+a_{10}x^7$	$+a_{10}x^6$	$+a_{10}x^5$	$+a_{10}x^4$		$+a_{10}x^2$	$+a_{10}x$	$+a_{10}$
a_9x^{19}	a_9x^9	$+a_9x^8$	$+a_9x^7$	$+a_9x^6$			$+a_9x^3$	$+a_9x^2$	$+a_9x$	$+a_9$
a_8x^{18}	a_8x^9			$+a_8x^6$		$+a_8x^4$			$+a_8x$	$+a_8$
a_7x^{17}	a_7x^9		$+a_7x^7$			$+a_7x^4$	$+a_7x^3$	$+a_7x^2$		$+a_7$
a_6x^{16}	a_6x^9		$+a_6x^7$	$+a_6x^6$	$+a_6x^5$	$+a_6x^4$	$+a_6x^3$		$+a_6x$	
a_5x^{15}		$+a_5x^8$		$+a_5x^6$	$+a_5x^5$	$+a_5x^4$	$+a_5x^3$	$+a_5x^2$		$+a_5$
a_4x^{14}	a_4x^9	$+a_4x^8$					$+a_4x^3$		$+a_4x$	
a_3x^{13}		$+a_3x^8$	$+a_3x^7$					$+a_3x^2$		
a_2x^{12}	a_2x^9	$+a_2x^8$		$+a_2x^6$	$+a_2x^5$	$+a_2x^4$		$+a_2x^2$	$+a_2x$	
a_1x^{11}		$+a_1x^8$	$+a_1x^7$		$+a_1x^5$	$+a_1x^4$	$+a_1x^3$		$+a_1x$	$+a_1$
a_0x^{10}	a_0x^9	$+a_0x^8$		$+a_0x^6$	$+a_0x^5$		$+a_0x^3$			$+a_0$
Залишок від ділення $A(x)\cdot x^{10}$ на $G(x)$	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓	⇓
$R(x)$	c_9x^9	$+c_8x^8$	$+c_7x^7$	$+c_6x^6$	$+c_5x^5$	$+c_4x^4$	$+c_3x^3$	$+c_2x^2$	$+c_1x$	$+c_0$

Рис. 40. Залишки від ділення $A(x)\cdot x^{10}$ на $G(x)$

$$c_5 = a_{20} + a_{18} + a_{16} + a_{14} + a_{11} + a_{10} + a_6 + a_5 + a_2 + a_1 + a_0 \pmod 2;$$

$$c_4 = a_{20} + a_{19} + a_{17} + a_{13} + a_{12} + a_{10} + a_8 + a_7 + a_6 + a_5 + a_2 + a_1 \pmod 2;$$

$$c_3 = a_{19} + a_{18} + a_{12} + a_{11} + a_9 + a_7 + a_6 + a_5 + a_4 + a_1 + a_0 \pmod 2;$$

$$c_2 = a_{20} + a_{18} + a_{15} + a_{12} + a_{11} + a_{10} + a_9 + a_7 + a_5 + a_3 + a_2 \pmod 2;$$

$$c_1 = a_{19} + a_{17} + a_{14} + a_{11} + a_{10} + a_9 + a_8 + a_6 + a_4 + a_2 + a_1 \pmod 2;$$

$$c_0 = a_{18} + a_{16} + a_{13} + a_{10} + a_9 + a_8 + a_7 + a_5 + a_3 + a_1 + a_0 \pmod 2.$$

Структуру коду БЧХ (31, 21) наведено на рис. 41, а схему формування коду — на рис. 42.

Код БЧХ (31, 21) — це код з виправленням подвійних помилок (DEC). Можливість виправлення зумовлена тим, що корінь вихідного многочлена α підноситься до декількох степенів, які створюють послідовний ряд цілих чисел (див. табл. 21). До останнього часу це твердження доводилося на прикладах інших кодів БЧХ, тут розглянемо його справедливість на прикладі коду БЧХ (31, 21).

Принцип формування контрольних розрядів коду (31, 21) з інформаційних розрядів продемонстровано на рис. 40, а контрольні розряди, якими відображається кожен інформаційний розряд, наведено в табл. 24. Як бачимо, мінімальна кодова відстань коду БЧХ (31, 21) дорівнює 5. Отже, за допомогою цього коду можна виправляти подвійні помилки (DEC).

Оскільки код БЧХ (31, 21) — це код DEC, то можна виявити усі кодові конфігурації з помилками в одному і двох розрядах.

Число кодових конфігурацій з помилками не більш ніж у двох розрядах визначається таким чином:

- помилок немає — $\binom{31}{0} = \frac{31!}{0!(31-0)!} = 1;$
- помилка в одному розряді — $\binom{31}{1} = \frac{31!}{0!(31-1)!} = 31;$

Інформаційні розряди																					Контрольні розряди										
a_{20}	a_{19}	a_{18}	a_{17}	a_{16}	a_{15}	a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	c_9	c_8	c_7	c_6	c_5	c_4	c_3	c_2	c_1	c_0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31

Розряди c_9—c_0 пов'язані формулою (29)

Рис. 41. Структура коду БЧХ (31, 21)

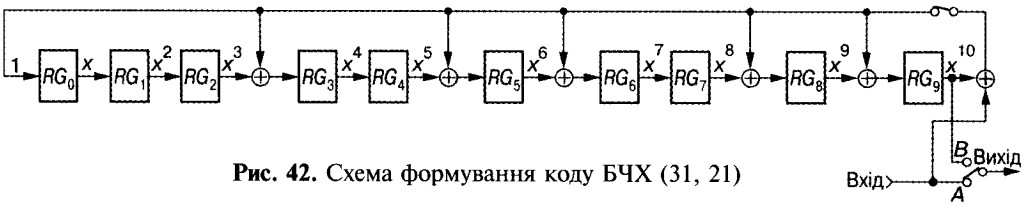


Рис. 42. Схема формування коду БЧХ (31, 21)

Таблиця 24. Інформаційні і контрольні розряди, що відображають їх

Інформаційний розряд	Контрольні розряди	Число розрядів, що змінюються у разі зміни інформаційного розряду
a_{20}	$c_9, c_8, c_7, c_5, c_4, c_2$	7
a_{19}	$c_8, c_7, c_6, c_4, c_3, c_1$	7
a_{18}	$c_7, c_8, c_5, c_3, c_2, c_0$	7
a_{17}	c_9, c_8, c_7, c_4, c_1	6
a_{16}	c_8, c_7, c_6, c_5, c_0	6
a_{15}	c_9, c_8, c_6, c_2	5
a_{14}	c_8, c_7, c_5, c_1	5
a_{13}	c_7, c_6, c_4, c_0	5
a_{12}	$c_9, c_8, c_7, c_6, c_4, c_3, c_2$	8
a_{11}	$c_8, c_7, c_6, c_5, c_3, c_2, c_1$	8
a_{10}	$c_7, c_6, c_5, c_4, c_2, c_1, c_0$	8
a_9	$c_9, c_8, c_7, c_6, c_3, c_2, c_1, c_0$	9
a_8	c_9, c_6, c_4, c_1, c_0	6
a_7	$c_9, c_7, c_4, c_3, c_2, c_0$	7
a_6	$c_9, c_7, c_6, c_5, c_4, c_3, c_1$	8
a_5	$c_8, c_6, c_5, c_4, c_3, c_2, c_0$	8
a_4	c_9, c_8, c_3, c_1	5
a_3	c_8, c_7, c_2, c_0	5
a_2	$c_9, c_8, c_6, c_5, c_4, c_2, c_1$	8
a_1	$c_8, c_7, c_5, c_4, c_2, c_1, c_0$	8
a_0	$c_9, c_8, c_6, c_5, c_3, c_0$	7

• помилки в двох розрядах — $\binom{31}{2} = \frac{31!}{0!(31-2)!} = 465.$

Загальне число кодових конфігурацій з помилками в цьому випадку становить

$$\binom{31}{0} + \binom{31}{1} + \binom{31}{2} = 1 + 31 + 465 = 497.$$

Водночас число контрольних розрядів коду БЧХ (31, 21) дорівнює 10, отже, цими контрольними розрядами можна відобразити $2^{10} = 1024$ кодових конфігурацій. Таким чином, загальне число кодових конфігурацій з помилками не більш ніж у двох розрядах (497) менше, ніж число кодових комбінацій, що відображаються десятьма контрольними розрядами (1024).

Оскільки число розрядів коду БЧХ (31, 21) — 31, то помилкові кодові конфігурації мають вигляд

$$e = (e_{30}, e_{29}, e_{28}, e_{27}, \ldots, e_1, e_0).$$

У кодовій конфігурації e_{30}—e_0 максимум два розряди містять "1".

Тепер можна визначити всі кодові конфігурації з помилками в одному і двох розрядах, які можуть бути виправлені.

Зауважимо, що кодові конфігурації з помилками в одному розряді належать моделі відстаней помилок (31). За допомогою однієї такої моделі можна виявити 31 кодову конфігурацію з помилкою в одному розряді. А оскільки кодових конфігурацій з помилкою в одному розряді всього 31, то вони виявляються всі.

Кодові конфігурації з помилками в двох розрядах відносяться до моделі відстаней помилок (j, k), де j і k задовольняють умови: $j + k = 31$, $j \le k$. Виявлення всіх моделей відстаней помилок (j, k) означає виявлення всіх кодових конфігурацій з помилками в двох розрядах. Таких моделей відстаней помилок є всього 15, а саме:

(1, 30), (2, 29), (3, 28), (4, 27), (5, 26), (6, 25), (7, 24), (8, 23),
(9, 22), (10, 21), (11, 20), (12, 19), (13, 18), (14, 17), (15, 16).

Отже, моделями відстаней помилок, що відображають помилки в двох розрядах, є наведені вище моделі від початкової (1, 30) до останньої (15, 16). У цих моделей відстаней помилок не рівні кодові відстані, тому однією моделлю можна виявити 31 кодову конфігурацію з помилками в двох розрядах. Отже, загальне число кодових конфігурацій з виявними помилками в двох розрядах становить $31 \cdot 15 = 465$, тобто за допомогою 15 моделей відстаней помилок (j, k) можуть бути виявлені всі кодові конфігурації з помилками в двох розрядах. Всі вони подані в табл. 25.

Таблиця 25. Модель відстані помилок і конфігурації виявних помилок

Модель відстані помилок	Конфігурації виявних помилок
(31)	(e_0), (e_1), (e_2), (e_3), (e_4), (e_5), (e_6), (e_7), (e_8), (e_9), (e_{10}), (e_{11}), (e_{12}), (e_{13}), (e_{14}), (e_{15}), (e_{16}), (e_{17}), (e_{18}), (e_{19}), (e_{20}), (e_{21}), (e_{22}), (e_{23}), (e_{24}), (e_{25}), (e_{26}), (e_{27}), (e_{28}), (e_{29}), (e_{30})
(1, 30)	(e_1, e_0), (e_2, e_1), (e_3, e_2), (e_4, e_3), (e_5, e_4), (e_6, e_5), (e_7, e_6), (e_8, e_7), (e_9, e_8), (e_{10}, e_9), (e_{11}, e_{10}), (e_{12}, e_{11}), (e_{13}, e_{12}), (e_{14}, e_{13}), (e_{15}, e_{14}), (e_{16}, e_{15}), (e_{17}, e_{16}), (e_{18}, e_{17}), (e_{19}, e_{18}), (e_{20}, e_{19}), (e_{21}, e_{20}), (e_{22}, e_{21}), (e_{23}, e_{22}), (e_{24}, e_{23}), (e_{25}, e_{24}), (e_{26}, e_{25}), (e_{27}, e_{26}), (e_{28}, e_{27}), (e_{29}, e_{28}), (e_{30}, e_{29}), (e_{30}, e_0)
(2, 29)	(e_2, e_0), (e_3, e_1), (e_4, e_2), (e_5, e_3), (e_6, e_4), (e_7, e_5), (e_8, e_6), (e_9, e_7), (e_{10}, e_8), (e_{11}, e_9), (e_{12}, e_{10}), (e_{13}, e_{11}), (e_{14}, e_{12}), (e_{15}, e_{13}), (e_{16}, e_{14}), (e_{17}, e_{15}), (e_{18}, e_{16}), (e_{19}, e_{17}), (e_{20}, e_{18}), (e_{21}, e_{19}), (e_{22}, e_{20}), (e_{23}, e_{21}), (e_{24}, e_{22}), (e_{25}, e_{23}), (e_{26}, e_{24}), (e_{27}, e_{25}), (e_{28}, e_{26}), (e_{29}, e_{27}), (e_{30}, e_{28}), (e_{29}, e_0), (e_{30}, e_1)
(3, 28)	(e_3, e_0), (e_4, e_1), (e_5, e_2), (e_6, e_3), (e_7, e_4), (e_8, e_5), (e_9, e_6), (e_{10}, e_7), (e_{11}, e_8), (e_{12}, e_9), (e_{13}, e_{10}), (e_{14}, e_{11}), (e_{15}, e_{12}), (e_{16}, e_{13}), (e_{17}, e_{14}), (e_{18}, e_{15}), (e_{19}, e_{16}), (e_{20}, e_{17}), (e_{21}, e_{18}), (e_{22}, e_{19}), (e_{23}, e_{20}), (e_{24}, e_{21}), (e_{25}, e_{22}), (e_{26}, e_{23}), (e_{27}, e_{24}), (e_{28}, e_{25}), (e_{29}, e_{26}), (e_{30}, e_{27}), (e_{28}, e_0), (e_{29}, e_1), (e_{30}, e_2)
(4, 27)	(e_4, e_0), (e_5, e_1), (e_6, e_2), (e_7, e_3), (e_8, e_4), (e_9, e_5), (e_{10}, e_6), (e_{11}, e_7), (e_{12}, e_8), (e_{13}, e_9), (e_{14}, e_{10}), (e_{15}, e_{11}), (e_{16}, e_{12}), (e_{17}, e_{13}), (e_{18}, e_{14}), (e_{19}, e_{15}), (e_{20}, e_{16}), (e_{21}, e_{17}), (e_{22}, e_{18}), (e_{23}, e_{19}), (e_{24}, e_{20}), (e_{25}, e_{21}), (e_{26}, e_{22}), (e_{27}, e_{23}), (e_{28}, e_{24}), (e_{29}, e_{25}), (e_{30}, e_{26}), (e_{27}, e_0), (e_{28}, e_1), (e_{29}, e_2), (e_{30}, e_3)
(5, 26)	(e_5, e_0), (e_6, e_1), (e_7, e_2), (e_8, e_3), (e_9, e_4), (e_{10}, e_5), (e_{11}, e_6), (e_{12}, e_7), (e_{13}, e_8), (e_{14}, e_9), (e_{15}, e_{10}), (e_{16}, e_{11}), (e_{17}, e_{12}), (e_{18}, e_{13}), (e_{19}, e_{14}), (e_{20}, e_{15}), (e_{21}, e_{16}), (e_{22}, e_{17}), (e_{23}, e_{18}), (e_{24}, e_{19}), (e_{25}, e_{20}), (e_{26}, e_{21}), (e_{27}, e_{22}), (e_{28}, e_{23}), (e_{29}, e_{24}), (e_{30}, e_{25}), (e_{26}, e_0), (e_{27}, e_1), (e_{28}, e_2), (e_{29}, e_3), (e_{30}, e_4)
(6, 25)	(e_6, e_0), (e_7, e_1), (e_8, e_2), (e_9, e_3), (e_{10}, e_4), (e_{11}, e_5), (e_{12}, e_6), (e_{13}, e_7), (e_{14}, e_8), (e_{15}, e_9), (e_{16}, e_{10}), (e_{17}, e_{11}), (e_{18}, e_{12}), (e_{19}, e_{13}), (e_{20}, e_{14}), (e_{21}, e_{15}), (e_{22}, e_{16}), (e_{23}, e_{17}), (e_{24}, e_{18}), (e_{25}, e_{19}), (e_{26}, e_{20}), (e_{27}, e_{21}), (e_{28}, e_{22}), (e_{29}, e_{23}), (e_{30}, e_{24}), (e_{25}, e_0), (e_{26}, e_1), (e_{27}, e_2), (e_{28}, e_3), (e_{29}, e_4), (e_{30}, e_5)

Закінчення табл. 25

Модель відстані помилок	Конфігурації виявних помилок
(7, 24)	(e_7, e_0), (e_8, e_1), (e_9, e_2), (e_{10}, e_3), (e_{11}, e_4), (e_{12}, e_5), (e_{13}, e_6), (e_{14}, e_7), (e_{15}, e_8), (e_{16}, e_9), (e_{17}, e_{10}), (e_{18}, e_{11}), (e_{19}, e_{12}), (e_{20}, e_{13}), (e_{21}, e_{14}), (e_{22}, e_{15}), (e_{23}, e_{16}), (e_{24}, e_{17}), (e_{25}, e_{18}), (e_{26}, e_{19}), (e_{27}, e_{20}), (e_{28}, e_{21}), (e_{29}, e_{22}), (e_{30}, e_{23}), (e_{24}, e_0), (e_{25}, e_1), (e_{26}, e_2), (e_{27}, e_3), (e_{28}, e_4), (e_{29}, e_5), (e_{30}, e_6)
(8, 23)	(e_8, e_0), (e_9, e_1), (e_{10}, e_2), (e_{11}, e_3), (e_{12}, e_4), (e_{13}, e_5), (e_{14}, e_6), (e_{15}, e_7), (e_{16}, e_8), (e_{17}, e_9), (e_{18}, e_{10}), (e_{19}, e_{11}), (e_{20}, e_{12}), (e_{21}, e_{13}), (e_{22}, e_{14}), (e_{23}, e_{15}), (e_{24}, e_{16}), (e_{25}, e_{17}), (e_{26}, e_{18}), (e_{27}, e_{19}), (e_{28}, e_{20}), (e_{29}, e_{21}), (e_{30}, e_{22}), (e_{23}, e_0), (e_{24}, e_1), (e_{25}, e_2), (e_{26}, e_3), (e_{27}, e_4), (e_{28}, e_5), (e_{29}, e_6), (e_{30}, e_7)
(9, 22)	(e_9, e_0), (e_{10}, e_1), (e_{11}, e_2), (e_{12}, e_3), (e_{13}, e_4), (e_{14}, e_5), (e_{15}, e_6), (e_{16}, e_7), (e_{17}, e_8), (e_{18}, e_9), (e_{19}, e_{10}), (e_{20}, e_{11}), (e_{21}, e_{12}), (e_{22}, e_{13}), (e_{23}, e_{14}), (e_{24}, e_{15}), (e_{25}, e_{16}), (e_{26}, e_{17}), (e_{27}, e_{18}), (e_{28}, e_{19}), (e_{29}, e_{20}), (e_{30}, e_{21}), (e_{22}, e_0), (e_{23}, e_1), (e_{24}, e_2), (e_{25}, e_3), (e_{26}, e_4), (e_{27}, e_5), (e_{28}, e_6), (e_{29}, e_7), (e_{30}, e_8)
(10, 21)	(e_{10}, e_0), (e_{11}, e_1), (e_{12}, e_2), (e_{13}, e_3), (e_{14}, e_4), (e_{15}, e_5), (e_{16}, e_6), (e_{17}, e_7), (e_{18}, e_8), (e_{19}, e_9), (e_{20}, e_{10}), (e_{21}, e_{11}), (e_{22}, e_{12}), (e_{23}, e_{13}), (e_{24}, e_{14}), (e_{25}, e_{15}), (e_{26}, e_{16}), (e_{27}, e_{17}), (e_{28}, e_{18}), (e_{29}, e_{19}), (e_{30}, e_{20}), (e_{21}, e_0), (e_{22}, e_1), (e_{23}, e_2), (e_{24}, e_3), (e_{25}, e_4), (e_{26}, e_5), (e_{27}, e_6), (e_{28}, e_7), (e_{29}, e_8), (e_{30}, e_9)
(11, 20)	(e_{11}, e_0), (e_{12}, e_1), (e_{13}, e_2), (e_{14}, e_3), (e_{15}, e_4), (e_{16}, e_5), (e_{17}, e_6), (e_{18}, e_7), (e_{19}, e_8), (e_{20}, e_9), (e_{21}, e_{10}), (e_{22}, e_{11}), (e_{23}, e_{12}), (e_{24}, e_{13}), (e_{25}, e_{14}), (e_{26}, e_{15}), (e_{27}, e_{16}), (e_{28}, e_{17}), (e_{29}, e_{18}), (e_{30}, e_{19}), (e_{20}, e_0), (e_{21}, e_1), (e_{22}, e_2), (e_{23}, e_3), (e_{24}, e_4), (e_{25}, e_5), (e_{26}, e_6), (e_{27}, e_7), (e_{28}, e_8), (e_{29}, e_9), (e_{30}, e_{10})
(12, 19)	(e_{12}, e_0), (e_{13}, e_1), (e_{14}, e_2), (e_{15}, e_3), (e_{16}, e_4), (e_{17}, e_5), (e_{18}, e_6), (e_{19}, e_7), (e_{20}, e_8), (e_{21}, e_9), (e_{22}, e_{10}), (e_{23}, e_{11}), (e_{24}, e_{12}), (e_{25}, e_{13}), (e_{26}, e_{14}), (e_{27}, e_{15}), (e_{28}, e_{16}), (e_{29}, e_{17}), (e_{30}, e_{18}), (e_{19}, e_0), (e_{20}, e_1), (e_{21}, e_2), (e_{22}, e_3), (e_{23}, e_4), (e_{24}, e_5), (e_{25}, e_6), (e_{26}, e_7), (e_{27}, e_8), (e_{28}, e_9), (e_{29}, e_{10}), (e_{30}, e_{11})
(13, 18)	(e_{13}, e_0), (e_{14}, e_1), (e_{15}, e_2), (e_{16}, e_3), (e_{17}, e_4), (e_{18}, e_5), (e_{19}, e_6), (e_{20}, e_7), (e_{21}, e_8), (e_{22}, e_9), (e_{23}, e_{10}), (e_{24}, e_{11}), (e_{25}, e_{12}), (e_{26}, e_{13}), (e_{27}, e_{14}), (e_{28}, e_{15}), (e_{29}, e_{16}), (e_{30}, e_{17}), (e_{18}, e_0), (e_{19}, e_1), (e_{20}, e_2), (e_{21}, e_3), (e_{22}, e_4), (e_{23}, e_5), (e_{24}, e_6), (e_{25}, e_7), (e_{26}, e_8), (e_{27}, e_9), (e_{28}, e_{10}), (e_{29}, e_{11}), (e_{30}, e_{12})
(14, 17)	(e_{14}, e_0), (e_{15}, e_1), (e_{16}, e_2), (e_{17}, e_3), (e_{18}, e_4), (e_{19}, e_5), (e_{20}, e_6), (e_{21}, e_7), (e_{22}, e_8), (e_{23}, e_9), (e_{24}, e_{10}), (e_{25}, e_{11}), (e_{26}, e_{12}), (e_{27}, e_{13}), (e_{28}, e_{14}), (e_{29}, e_{15}), (e_{30}, e_{16}), (e_{17}, e_0), (e_{18}, e_1), (e_{19}, e_2), (e_{20}, e_3), (e_{21}, e_4), (e_{22}, e_5), (e_{23}, e_6), (e_{24}, e_7), (e_{25}, e_8), (e_{26}, e_9), (e_{27}, e_{10}), (e_{28}, e_{11}), (e_{29}, e_{12}), (e_{30}, e_{13})
(15, 16)	(e_{15}, e_0), (e_{16}, e_1), (e_{17}, e_2), (e_{18}, e_3), (e_{19}, e_4), (e_{20}, e_5), (e_{21}, e_6), (e_{22}, e_7), (e_{23}, e_8), (e_{24}, e_9), (e_{25}, e_{10}), (e_{26}, e_{11}), (e_{27}, e_{12}), (e_{28}, e_{13}), (e_{29}, e_{14}), (e_{30}, e_{15}), (e_{16}, e_0), (e_{17}, e_1), (e_{18}, e_2), (e_{19}, e_3), (e_{20}, e_4), (e_{21}, e_5), (e_{22}, e_6), (e_{23}, e_7), (e_{24}, e_8), (e_{25}, e_9), (e_{26}, e_{10}), (e_{27}, e_{11}), (e_{28}, e_{12}), (e_{29}, e_{13}), (e_{30}, e_{14})

Після визначення моделей відстаней помилок знаходять залишки від їх ділення, тобто синдроми моделей відстаней помилок. Розглянемо практичну реалізацію цього процесу.

Залишок від ділення моделей відстаней помилок з помилкою в одному розряді записується так:

$$R(\alpha) = 1, \tag{30}$$

а залишок від ділення моделей відстаней помилок з помилками в двох розрядах —

$$R(\alpha) = \alpha^j + 1. \quad (31)$$

Оскільки α є коренем породжувального многочлена $G(x)$, який визначається формулою

$$G(x) = x^{10} + x^9 + x^8 + x^6 + x^5 + x^3 + 1,$$

то має місце рівність

$$G(\alpha) = 0,$$

тобто

$$\alpha^{10} = \alpha^9 + \alpha^8 + \alpha^6 + \alpha^5 + \alpha^3 + 1. \quad (32)$$

Отже, якщо корінь α у залишку від ділення моделей відстаней помилок, що визначається формулою (31), має степінь вищий ніж α^{10}, то необхідно зробити так, щоб степінь залишку став менший, ніж степінь породжувального многочлена. Для пониження степеня α потрібно скористатися співвідношенням (32).

Залишок від ділення моделей відстаней помилок з помилками в двох розрядах, що задається формулою (31), містять такі доданки з α в степені вище ніж 10:

$$\alpha^{10}, \alpha^{11}, \alpha^{12}, \alpha^{13}, \alpha^{14}, \alpha^{15}.$$

Для визначення синдрому їх можна замінити доданками з α в степені не більше ніж 9 за допомогою рівностей типу (32):

$$\alpha^{10} = \alpha^9 + \alpha^8 + \alpha^6 + \alpha^5 + \alpha^3 + 1,$$
$$\alpha^{11} = \alpha^8 + \alpha^7 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha + 1,$$
$$\alpha^{12} = \alpha^9 + \alpha^8 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^2 + \alpha,$$
$$\alpha^{13} = \alpha^8 + \alpha^7 + \alpha^2 + 1,$$
$$\alpha^{14} = \alpha^9 + \alpha^8 + \alpha^3 + \alpha,$$
$$\alpha^{11} = \alpha^8 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + 1.$$

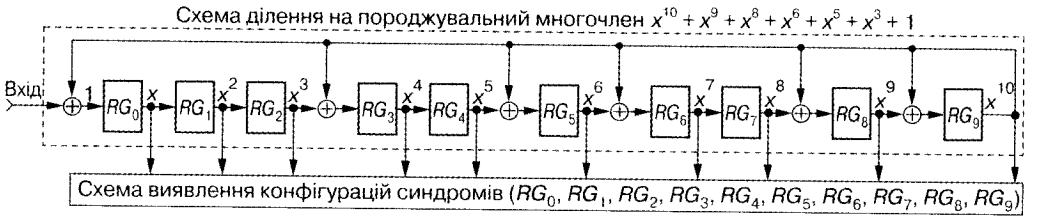
Ці формули є основоположними у разі визначення синдромів для аналізу моделей відстаней помилок. Моделі відстаней помилок, залишки від їх ділення і відповідні їм конфігурації синдромів наведені на рис. 43.

Схеми декодування моделей відстаней помилок коду БЧХ (31, 21) і виправлення помилок у двох розрядах подано на рис. 44.

Як приклад розглянемо роботу цієї схеми при помилках в 26-му (c_5) і 16-му (a_5) розрядах, тобто помилковою є конфігурація (e_{15}, e_5). Це означає, що розряди $e_{15} = 1$ і $e_5 = 1$, а решта розрядів дорівнюють "0". Отже, 16-й і 26-й біти інформаційної послідовності, що приймається, мають такі значення:

16-й біт: $a_5 = a_5 + e_5$

26-й біт: $c_5 = c_5 + e_5$



Модель відстаней помилок	Залишок моделі відстаней помилок	Конфігурація синдрому, що виявляє модель відстаней помилок									
		(RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7	RG_8	RG_9)
(31)	$R(\alpha) = 1$	(1	0	0	0	0	0	0	0	0	0)
(1, 30)	$R(\alpha) = 1 + \alpha$	(1	1	0	0	0	0	0	0	0	0)
(2, 29)	$R(\alpha) = 1 + \alpha^2$	(1	0	1	0	0	0	0	0	0	0)
(3, 28)	$R(\alpha) = 1 + \alpha^3$	(1	0	0	1	0	0	0	0	0	0)
(4, 27)	$R(\alpha) = 1 + \alpha^4$	(1	0	0	0	1	0	0	0	0	0)
(5, 26)	$R(\alpha) = 1 + \alpha^5$	(1	0	0	0	0	1	0	0	0	0)
(6, 25)	$R(\alpha) = 1 + \alpha^6$	(1	0	0	0	0	0	1	0	0	0)
(7, 24)	$R(\alpha) = 1 + \alpha^7$	(1	0	0	0	0	0	0	1	0	0)
(8, 23)	$R(\alpha) = 1 + \alpha^8$	(1	0	0	0	0	0	0	0	1	0)
(9, 22)	$R(\alpha) = 1 + \alpha^9$	(1	0	0	0	0	0	0	0	0	1)
(10, 21)	$R(\alpha) = 1 + \alpha^{10} = \alpha^3 + \alpha^5 + \alpha^6 + \alpha^8 + \alpha^9$	(0	0	0	1	0	1	1	0	1	1)
(11, 20)	$R(\alpha) = 1 + \alpha^{11} = \alpha + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^7 + \alpha^8$	(0	1	0	1	1	1	0	1	1	0)
(12, 19)	$R(\alpha) = 1 + \alpha^{12} = 1 + \alpha + \alpha^2 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^8 + \alpha^9$	(1	1	1	0	1	1	1	0	1	1)
(13, 18)	$R(\alpha) = 1 + \alpha^{13} = \alpha^2 + \alpha^7 + \alpha^8$	(0	0	1	0	0	0	0	1	1	0)
(14, 17)	$R(\alpha) = 1 + \alpha^{14} = 1 + \alpha + \alpha^3 + \alpha^8 + \alpha^9$	(1	1	0	1	0	0	0	0	1	1)
(15, 16)	$R(\alpha) = 1 + \alpha^{15} = \alpha^2 + \alpha^3 + \alpha^4 + \alpha^5 + \alpha^6 + \alpha^8$	(0	0	1	1	1	1	1	0	1	0)

Рис. 43. Модель відстаней помилок і відповідні ним конфігурації синдромів

Проаналізуємо роботу схеми декодування з того моменту, коли вона, прийнявши кодову послідовність з помилковою конфігурацією (e_{15}, e_5), повністю визначила залишок від ділення помилкової конфігурації (e_{15}, e_5), тобто

$$R(\alpha) = e_{15}\alpha^{15} + e_5\alpha^5 = e_{15}(\alpha^8 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + 1) + e_5\alpha^5 =$$
$$= e_{15}\alpha^8 + e_{15}\alpha^6 + (e_{15} + e_5)\alpha^5 + e_{15}\alpha^4 + e_{15}\alpha^3 + e_{15}\alpha^2 + e_{15},$$

і зображена на рис. 44 схема декодування моделей відстаней помилок коду БЧХ (31, 21) починає працювати так, як це показано в табл. 26.

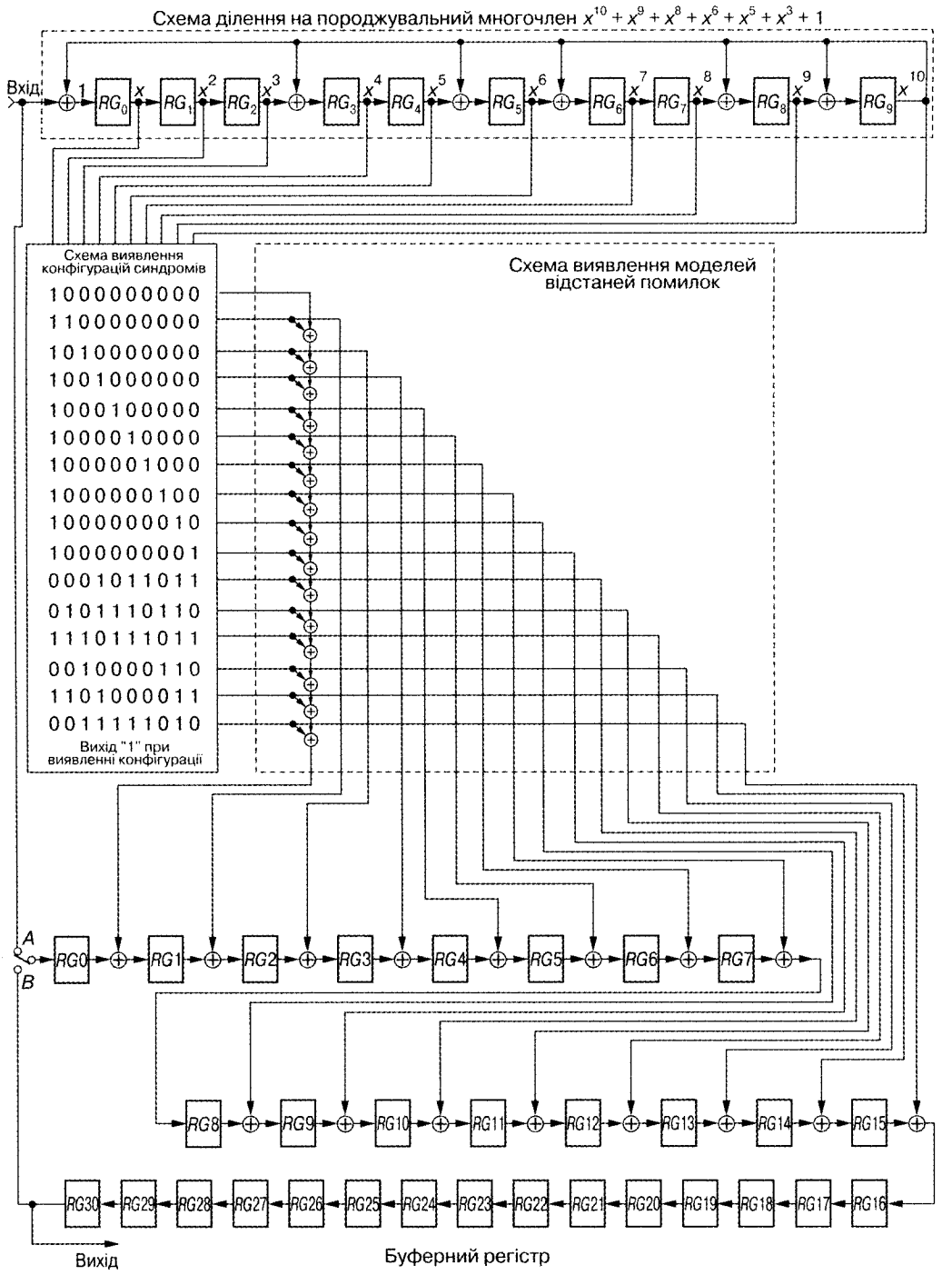


Рис. 44. Схема декодування моделей відстаней помилок коду БЧХ (31, 21)

Таблиця 26. Приклад виправлення

Перемикач	Такт	Схема ділення на породжувальний многочлен									
		RG_0	RG_1	RG_2	RG_3	RG_4	RG_5	RG_6	RG_7	RG_8	RG_9
B	0	e_{15}	0	e_{15}	e_{15}	e_{15}	$e_{15} + e_5$	e_{15}	0	e_{15}	0
	1	0	e_{15}	0	e_{15}	e_{15}	e_{15}	$e_{15} + e_5$	e_{15}	0	e_{15}
	2	e_{15}	0	e_{15}	e_{15}	e_{15}	0	0	$e_{15} + e_5$	0	e_{15}
	3	e_{15}	e_{15}	0	0	e_{15}	0	e_{15}	0	e_5	e_{15}
	4	e_{15}	e_{15}	e_{15}	e_{15}	0	0	e_{15}	e_{15}	e_{15}	$e_{15} + e_5$
	5	$e_{15} + e_5$	e_{15}	e_{15}	e_5	e_{15}	$e_{15} + e_5$	$e_{15} + e_5$	e_{15}	e_5	e_5
	6	e_5	$e_{15} + e_5$	e_{15}	$e_{15} + e_5$	e_5	$e_{15} + e_5$	e_{15}	$e_{15} + e_5$	$e_{15} + e_5$	0
	7	0	e_5	$e_{15} + e_5$	e_{15}	$e_{15} + e_5$	e_5	$e_{15} + e_5$	e_{15}	$e_{15} + e_5$	$e_{15} + e_5$
	8	$e_{15} + e_5$	0	e_5	0	e_{15}	0	e_{15}	$e_{15} + e_5$	e_5	0
	9	0	$e_{15} + e_5$	0	e_5	0	e_{15}	0	e_{15}	$e_{15} + e_5$	e_5
	10	e_5	0	$e_{15} + e_5$	e_5	e_5	e_5	$e_{15} + e_5$	0	$e_{15} + e_5$	e_{15}
	11	e_{15}	e_5	0	e_5	e_5	$e_{15} + e_5$	$e_{15} + e_5$	$e_{15} + e_5$	e_{15}	e_5
	12	e_5	e_{15}	e_5	e_5	e_5	0	e_{15}	$e_{15} + e_5$	e_{15}	$e_{15} + e_5$
	13	$e_{15} + e_5$	e_5	e_{15}	e_{15}	e_5	e_{15}	$e_{15} + e_5$	e_{15}	0	e_5
	14	e_5	$e_{15} + e_5$	e_5	$e_{15} + e_5$	e_{15}	0	$e_{15} + e_5$	$e_{15} + e_5$	$e_{15} + e_5$	e_5
	15	e_5	e_5	$e_{15} + e_5$	0	$e_{15} + e_5$	$e_{15} + e_5$	e_5	$e_{15} + e_5$	e_{15}	e_{15}
	16	e_{15}	e_5	e_5	e_5	0	e_5	e_5	e_5	e_5	0
	17	0	e_{15}	e_5	e_5	e_5	0	e_5	e_5	e_5	e_5
	18	e_5	0	e_{15}	0	e_5	0	e_5	e_5	0	0
	19	0	e_5	0	e_{15}	0	e_5	0	e_5	e_5	0
	20	0	0	e_5	0	e_{15}	0	e_5	0	e_5	e_5
	21	e_5	0	0	0	0	$e_{15} + e_5$	e_5	e_5	e_5	0
	22	0	e_5	0	0	0	0	$e_{15} + e_5$	e_5	e_5	e_5
	23	e_5	0	e_5	e_5	0	e_5	e_5	$e_{15} + e_5$	0	0
	24	0	e_5	0	e_5	e_5	0	e_5	e_5	$e_{15} + e_5$	0
	25	0	0	e_5	0	e_5	e_5	0	e_5	e_5	$e_{15} + e_5$
	26	$e_{15} + e_5$	0	0	e_{15}	0	e_{15}	e_{15}	0	e_{15}	e_{15}
	27	e_{15}	$e_{15} + e_5$	0	e_{15}	e_{15}	e_{15}	0	e_{15}	e_{15}	0
	28	0	e_{15}	$e_{15} + e_5$	0	e_{15}	e_{15}	e_{15}	0	e_{15}	e_{15}
	29	e_{15}	0	e_{15}	e_{15}	0	0	0	e_{15}	e_{15}	0
	30	0	e_{15}	0	e_{15}	e_5	0	0	0	e_{15}	e_{15}
	31	—	—	—	—	—	—	—	—	—	—
	32	—	—	—	—	—	—	—	—	—	—
A	33	—	—	—	—	—	—	—	—	—	—
		⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
		—	—	—	—	—	—	—	—	—	—

e_{15}, $e_5 = 1$, інші дорівнюють "0" $\quad \overline{a}_5 = a_5 + e_{15}$, $\overline{c}_5 = c_5 + e_5$

подвійних помилок

Вхід схеми виявлення конфігурації синдрому	Приймальний буферний регістр зсуву														Вихід
	*RG*0	*RG*1	*RG*2	*RG*3	*RG*4	*RG*5	*RG*6	*RG*7	*RG*8	*RG*9	*RG*10	*RG*11	...	*RG*30	
(1011101010)	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	...	a_{20}	—
(0101110101)	a_{20}	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	...	a_{19}	—
(1011100001)	a_{19}	a_{20}	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	...	a_{18}	—
(1100101011)	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	...	a_{17}	—
(1111001110)	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	...	a_{16}	—
(0111100111)	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	...	a_{15}	—
(1111001110)	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	...	a_{14}	—
(0101010100)	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	c_3	c_4	...	a_{13}	—
(0010101010)	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	c_3	...	a_{12}	—
(0001010101)	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	c_2	...	a_{11}	—
(1001110001)	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	c_1	...	a_{10}	—
(1101100011)	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	c_0	...	a_9	—
(1111101010)	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	a_{20}	...	a_8	—
(0111110101)	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	a_{19}	...	a_7	—
(1010100001)	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{18}	...	a_6	—
(1100001011)	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	...	a_5	—
(1111011110)	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	...	a_4	—
(0111101111)	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	...	a_3	—
(1010101100)	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	a_{14}	...	a_2	—
(0101010110)	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	a_{13}	...	a_1	—
(0010101011)	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	a_{11}	a_{12}	...	a_0	—
(1000001110)	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	a_{11}	...	c_9	—
(0100000111)	c_9	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	a_{10}	...	c_8	—
(1011011000)	c_8	c_9	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	a_9	...	c_7	—
(0101101100)	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	a_8	...	c_6	—
(0010110110)	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	a_7	...	c_5	—
(0001011011)	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	a_6	...	c_4	—
Конфігурація синдрому для моделі відстаней помилок (10, 21)	→ Вихід схеми виявлення моделей відстаней помилок														
→ Вихід виявленої конфігурації = 1	1	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	(0)	1	(0)	(0)	
(0001011011)	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	$\bar{a}_5$	...	c_3	—
(0100111011)	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	a_4	...	c_2	—
(1011000110)	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	a_2	a_3	...	c_1	—
(0101100011)	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	a_2	...	c_0	—
—	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	a_1	...	a_{20}	—
—	—	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	a_0	...	a_{19}	—
—	—	—	c_0	c_1	c_2	c_3	c_4	$\bar{c}_5$	c_6	c_7	c_8	c_9	...	a_{18}	—
—	—	—	—	—	—	—	—	—	—	—	—	—	...	c_0	—

СПОСІБ СТВОРЕННЯ ЦИКЛІЧНОГО КОДУ

Код БЧХ аналогічний циклічному коду. У цьому розділі розглядаються питання створення різних циклічних кодів і їх структура без варіантів практичної реалізації кодерів і декодерів. Щодо можливості виправлення помилок, то перевіряти це, враховуючи аналогію з кодами БЧХ, немає жодної потреби.

Проте у разі самостійного створення циклічного коду все-таки є необхідність перевірити виправляючу здатність коду.

Під можливістю виправлення помилок мають на увазі мінімальну кодову відстань у всьому коді. Проте у разі збільшення числа інформаційних бітів дослідження кодової відстані по всьому кодовому полю стає досить складним.

Отже, що стосується можливості виправлення помилок, то у разі зміни числа інформаційних бітів коду на 1 біт необхідно таки виконувати перевірку на мінімальну кодову відстань.

19.1. СКЛАДНИЙ МНОГОЧЛЕН І ПЕРІОД ЦИКЛІЧНОГО КОДУ

При створенні коду БЧХ важливим є вихідний (початковий) многочлен. Позначимо степінь початкового многочлена $P(x)$ через M, тоді основа вихідного многочлена α має період $2M-1$, тобто

$$\alpha^{2M-1} = 1.$$

Тепер розглянемо многочлен $F(x)$:

$$F(x) = x^{2M-1} + 1.$$

Введемо до нього основу α вихідного многочлена $P(x)$:

$$F(\alpha) = \alpha^{2M-1} + 1 = 0.$$

Цей вираз має той сенс, що многочлен $x^{2M-1} + 1$ ділиться без залишку на вихідний многочлен $P(x)$. Цей вихідний многочлен є окремим випадком простих многочленів. Основа γ простого многочлена також має період, який у випадку $F(x) = x^n + 1$ задається найменшим показником степеня n:

$$F(\gamma) = \gamma^n + 1 = 0.$$

Таким чином, період простого многочлена можна визначити найменшим показником степеня n многочлена $x^n + 1$, який ділиться без залишку на простий многочлен.

Спосіб створення циклічного коду

Складний многочлен циклічного коду задається добутком простих многочленів. Код БЧХ також є циклічним кодом.

Тепер дослідимо многочлен, який утворює код БЧХ. Наприклад, розглянемо многочлен, за допомогою якого утворюється код БЧХ (15, 5). Цей многочлен раніше розглядався як добуток мінімальних многочленів при послідовному піднесенні до степеня основи α вихідного многочлена

$$x^4 + x + 1$$

для α, α^2, α^3, α^4, α^5, α^6,
тобто
для α, α^2, α^4

$$x^4 + x + 1,$$

для α^3, α^6

$$x^4 + x^3 + x^2 + x + 1,$$

для α^5

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

Запишемо добуток поліномів $G(x)$:

$$G(x) = (x^2 + x + 1)(x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1).$$

Складові многочлену коду БЧХ (три поліноми) є простими многочленами. Тепер визначимо періоди цих трьох простих многочленів, для цього скористаємося методом знаходження частки від ділення $x^n + 1$ на простий многочлен. Наприклад, знайдемо період многочлена $x^2 + x + 1$:

$$\begin{array}{r} x^{n-2} \\ \hline x^2 + x + 1 \overline{)\, x^n \qquad\qquad + 1} \\ \underline{x^n + x^{n-1} + x^{n-2}} \\ x^{n-1} + x^{n-2} \quad +1 \end{array}$$

Розглянемо вираз $x^{n-1} + x^{n-2} + 1$ і, на підставі того ділиться без залишку $x^{n-1} + x^{n-2} + 1$ на $x^2 + x + 1$, чи ні, перевіримо, чи задовольняє n співвідношення

$$x^2 = x^{n-1},$$

$$x = x^{n-2},$$

а саме:

$$2 = n - 1 \rightarrow n = 3,$$

$$1 = n - 2 \rightarrow n = 3.$$

Отже, при $n = 3$ многочлен $x^3 + 1$ ділиться без залишку на $x^2 + x + 1$, тобто період $x^2 + x + 1$ дорівнює 3. Аналогічно знаходимо період многочлена $x^4 + x^3 + x^2 + x + 1$:

$$\begin{array}{r} x^{n-4} \\ \hline x^4 + x^3 + x^2 + x + 1 \overline{)\, x^n \qquad\qquad\qquad + 1} \\ \underline{x^n + x^{n-1} + x^{n-2} + x^{n-3} + x^{n-4}} \\ x^{n-1} + x^{n-2} + x^{n-3} + x^{n-4} + 1 \end{array}$$

Тут перевіряючи, чи є залишком вираз $x^4 + x^3 + x^2 + x + 1$, з'ясовуємо чи ділиться многочлен $x^n + 1$ без залишку на $x^4 + x^3 + x^2 + x + 1$. Для цього, обчислюючи показник степеня n кожного x, потрібно визначити, чи збігатимуться ці значення n.

Із $x^4 = x^{n-1}$, $4 = n - 1 \rightarrow n = 5$.

Із $x^3 = x^{n-2}$, $3 = n - 2 \rightarrow n = 5$.

Із $x^2 = x^{n-3}$, $2 = n - 3 \rightarrow n = 5$.

Із $x = x^{n-4}$, $1 = n - 4 \rightarrow n = 5$.

Отже, при $n = 5$ многочлен $x^n + 1$ ділиться без залишку на многочлен $x^4 + x^3 + x^2 + x + 1$, а період $x^4 + x^3 + x^2 + x + 1$ дорівнює 5.

Щодо періоду многочлена $x^4 + x + 1$, то він є вихідним многочленом, тому його період — $2^4 - 1 = 15$ — визначається досить просто. Водночас цей період можна знайти, поділивши $x^n + 1$ на $x^4 + x + 1$. Таке ділення реалізовано за схемою, наведеною на рис. 45.

Для визначення періоду простого многочлена застосовують спосіб ділення $x^n + 1$ на простий многочлен; при цьому зі зростанням степеня многочлена розрахунок є дедалі складнішим. Найпростіший метод визначення періоду простого многочлена — це розрахунок періоду вихідного многочлена.

Нехай α — основа вихідного многочлена $x^4 + x + 1$, тоді

$$x^{15} + 1 = 0.$$

Основа $x^{15} + 1$ має 15 визначених значень:

$$\alpha, \alpha^2, \alpha^3, \alpha^4, \alpha^5, \alpha^6, \alpha^7, \alpha^8, \alpha^9, \alpha^{10}, \alpha^{11}, \alpha^{12}, \alpha^{13}, \alpha^{14}, \alpha^{15}.$$

Ці значення (піднесена послідовно до степеня основа α) утворюють функцію основ $x^{15} + 1$:

$$F(x) = x^{15} + 1,$$

де α набуває послідовно значень від α^2 до α^{15}, наприклад:

$$F(\alpha^2) = (\alpha^2)^{15} + 1 = (\alpha^{15})^2 + 1 = 0,$$

$$F(\alpha^3) = (\alpha^3)^{15} + 1 = (\alpha^{15})^3 + 1 = 0,$$

$$F(\alpha^4) = (\alpha^4)^{15} + 1 = (\alpha^{15})^4 + 1 = 0,$$

.....

Для цих визначених основ є п'ять мінімальних многочленів, а саме:

для α, α^2, α^4, α^8	$x^4 + x + 1$,
для α^3, α^6, α^9, α^{12}	$x^4 + x^3 + x^2 + x + 1$,
для α^5, α^{10}	$x^2 + x + 1$,
для α^7, α^{11}, α^{13}, α^{14}	$x^4 + x^3 + 1$,
для α^{15}	$x + 1$.

$$
\begin{array}{l}
x^{n-4} + x^{n-7} + x^{n-8} + x^{n-10} + x^{n-12} + x^{n-13} + x^{n-14} \\
\hline
x^4 + x + 1)\ x^n \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 1 \\
\quad x^n + x^{n-3} + x^{n-4} \\
\quad \hline
\quad\quad x^{n-3} + x^{n-4} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 1 \\
\quad\quad\quad x^{n-3} + x^{n-6} + x^{n-7} \\
\quad\quad\quad \hline
\quad\quad\quad\quad x^{n-4} + x^{n-6} + x^{n-7} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 1 \\
\quad\quad\quad\quad x^{n-4} + x^{n-7} + x^{n-8} \\
\quad\quad\quad\quad \hline
\quad\quad\quad\quad\quad x^{n-6} + x^{n-8} \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad + 1 \\
\quad\quad\quad\quad\quad x^{n-6} + x^{n-9} + x^{n-10} \\
\quad\quad\quad\quad\quad \hline
\quad\quad\quad\quad\quad\quad x^{n-8} + x^{n-9} + x^{n-10} \qquad\qquad\qquad\qquad\qquad\qquad + 1 \\
\quad\quad\quad\quad\quad\quad x^{n-8} + x^{n-11} + x^{n-12} \\
\quad\quad\quad\quad\quad\quad \hline
\quad\quad\quad\quad\quad\quad\quad x^{n-9} + x^{n-10} + x^{n-11} + x^{n-12} \qquad\qquad\qquad + 1 \\
\quad\quad\quad\quad\quad\quad\quad x^{n-9} + x^{n-12} + x^{n-13} \\
\quad\quad\quad\quad\quad\quad\quad \hline
\quad\quad\quad\quad\quad\quad\quad\quad x^{n-10} + x^{n-11} + x^{n-13} \qquad\qquad + 1 \\
\quad\quad\quad\quad\quad\quad\quad\quad x^{n-10} + x^{n-13} + x^{n-14} \\
\quad\quad\quad\quad\quad\quad\quad\quad \hline
\quad\quad\quad\quad\quad\quad\quad\quad\quad x^{n-11} + x^{n-14} + 1
\end{array}
$$

$$
\begin{cases} 4 = n - 3 \to n = 7 \\ 1 = n - 4 \to n = 5 \end{cases}
\qquad
\begin{cases} 4 = n - 6 \to n = 10 \\ 1 = n - 8 \to n = 9 \end{cases}
\qquad
\begin{cases} 4 = n - 11 \to n = 15 \\ 1 = n - 14 \to n = 15 \end{cases}
$$

↓

При $n = 15$ $x^n + 1$ ділиться без залишку на $x^4 + x + 1$

↓

Період $x^4 + x + 1$ дорівнює 15

Рис. 45. Схема порядку визначення періоду многочлена $x^4 + x + 1$

Кожний з цих мінімальних многочленів є простим. Отже, $x^{15} + 1$ можна розкласти на п'ять простих співмножників:

$$x^{15} + 1 = (x + \alpha)(x + \alpha^2)\ldots (x + \alpha^{15}) =$$
$$= (x + 1)(x^2 + x + 1)(x^4 + x + 1)(x^4 + x^3 + 1)(x^4 + x^3 + x^2 + x + 1).$$

На ці прості співмножники функція основ $x^{15} + 1$ ділиться без залишку, тому і період простих співмножників ділиться без залишку на 15, тобто діль-

ник періоду простих співмножників многочлена $x^{15} + 1$ дорівнює 15. Крім того, ці прості многочлени є многочленами четвертого степеня, зокрема наступні три із п'яти мінімальних многочленів:

$$x^4 + x + 1,$$
$$x^4 + x^3 + x^2 + x + 1,$$
$$x^4 + x^3 + 1.$$

У двох з них періоди простих многочленів дорівнюють 15 та 5:

$$x^4 + x + 1 \rightarrow \text{період } 15,$$
$$x^4 + x^3 + x^2 + x + 1 \rightarrow \text{період } 5.$$

Між періодами цих двох простих многочленів існує співвідношення, що називається дільником: 5 і 15 — це $2^4 - 1(=15)$ дільник. Зважаючи на це, періоди простих многочленів четвертого степеня визначають із рівності $2^4 - 1 = 15$, а далі знаходять усі дільники числа 15, а саме: 15, 5, 3, 1. Період простого многочлена четвертого степеня дорівнює будь-якому з цих чисел. Проте, і 3, і 1 не є періодами, оскільки вони менші ніж 4. Отже, серед дільників числа 15 як період простого многочлена четвертого степеня можна використати тільки числа 15 і 5.

Із усіх многочленів четвертого степеня тепер визначимо період многочлена $x^4 + x^3 + 1$. Очевидно, що $x^{15} + 1$ ділиться без залишку на $x^4 + x^3 + 1$. Тому необхідно перевірити, чи ділиться $x^5 + 1$ без залишку на $x^4 + x^3 + 1$:

$$\begin{array}{r|l} & x \\ \hline x^4 + x^3 + 1 & x^5 \qquad\qquad\quad + 1 \\ & x^5 + x^4 \qquad + x \\ \hline & x^4 \qquad\quad + x + 1 \end{array}$$

Як бачимо, $x^5 + 1$ не ділиться без залишку на $x^4 + x^3 + 1$, отже, період многочлена $x^4 + x^3 + 1$ дорівнює 15.

Виходячи з наведеного вище, коли степінь простого многочлена дорівнює m, то на нього без залишку ділиться многочлен $x^{2^m-1} + 1$. Це свідчить про те, що простий многочлен степеня m має період $n = 2^m - 1$.

Перевіримо тепер, чи є цей період мінімальним. Для цього розкладемо $2^m - 1$ на множники:

$$n = 2^m - 1 = n_1 \cdot n_2 \cdot \ldots \cdot n_k$$

і утворимо многочлен $x^{n_i} + 1$, що складається із співмножників зі степенями більшими, ніж степінь простого многочлена. З'ясуємо, чи ділиться без залишку многочлен $x^{n_i} + 1$ на простий многочлен. Якщо ділиться, то мінімальне значення n_i — період простого многочлена. Якщо многочлен $x^{n_i} + 1$, що складається із множників $n_i (n = 2^m - 1)$, ділиться без залишку на простий многочлен, то період — $n = 2^m - 1$.

Покажемо, як $x^n + 1$ ділиться без залишку на $x^{n_i} + 1$. Наприклад, при $n = 15$ отримаємо такі співмножники: $n = 5 \cdot 3$. Далі поділимо многочлен $x^{15} + 1$

відповідно на $x^5 + 1$ і $x^3 + 1$:

$$
\begin{array}{r}
x^{10} + x^5 + 1 \\
x^5 + 1 \overline{)\, x^{15} \qquad\quad + 1} \\
\underline{x^{15} + x^{10}} \\
x^{10} + x^5 + 1 \\
\underline{x^{10} + x^5 + 1} \\
0
\end{array}
$$

$$
\begin{array}{r}
x^{12} + x^9 + x^6 + x^3 + 1 \\
x^3 + 1 \overline{)\, x^{15} \qquad\quad + 1} \\
\underline{x^{15} + x^{12}} \\
x^{12} \qquad\quad + 1 \\
\underline{x^{12} + x^9} \\
x^9 \qquad\quad + 1 \\
\underline{x^9 + x^6} \\
x^6 \qquad\quad + 1 \\
\underline{x^6 + x^3} \\
x^3 + 1 \\
\underline{x^3 + 1} \\
0
\end{array}
$$

Тепер так само знайдемо періоди простих многочленів, що утворюють складний многочлен коду БЧХ (15, 5). Складний многочлен $G(x)$ — це добуток трьох простих поліномів, а саме:

$$G(x) = (x^2 + x + 1)(x^4 + x + 1)(x^4 + x^3 + x^2 + x + 1).$$

Визначимо період многочлена $x^2 + x + 1$. Оскільки степінь цього простого многочлена дорівнює 2, то його період становить $2^2 - 1 = 3$. Оскільки дільниками числа 3 є тільки 3 і 1, то період наведеного тричлена дорівнює 3. Далі поділимо вираз $x^3 + 1$ на $x^2 + x + 1$:

$$
\begin{array}{r}
x + 1 \\
x^2 + x + 1 \overline{)\, x^3 \qquad + 1} \\
\underline{x^3 + x^2 + x} \\
x^2 + x + 1 \\
\underline{x^2 + x + 1} \\
0
\end{array}
$$

Оскільки $x^3 + 1 = (x + 1)(x^2 + x + 1)$, то $x^3 + 1$ ділиться на $x^2 + x + 1$ без залишку.

Період наступного простого многочлена $x^4 + x + 1$ — це $2^4 - 1 = 15$. Співмножники, що дають у добутку 15 — це 5 і 3. Найвищий степінь много-

члена дорівнює 4, тому необхідно перевірити, чи є періодом число 5. Для цього поділимо поліном $x^5 + 1$ на $x^4 + x + 1$:

$$\begin{array}{r|l} & x \\ \hline x^4 + x + 1 & x^5 \qquad\qquad + 1 \\ & x^5 \quad + x^2 + x \\ \hline & x^2 + x + 1 \end{array}$$

Як бачимо, $x^5 + 1$ не ділиться без залишку на $x^4 + x + 1$. Отже, число 5 не є періодом і як період слід використовувати число 15.

Нарешті, оскільки степінь простого многочлена $x^4 + x^3 + x^2 + x + 1$ дорівнює 4, то його період — $2^4 - 1 = 15$. Добуток співмножників становить 15 — це 5 і 3, але, оскільки 3 менше, ніж степінь даного простого многочлена, слід перевірити, чи є періодом число 5. Для цього поділимо многочлен $x^5 + 1$ на $x^4 + x^3 + x^2 + x + 1$:

$$\begin{array}{r|l} & x + 1 \\ \hline x^4 + x^3 + x^2 + x + 1 & x^5 \qquad\qquad\qquad + 1 \\ & x^5 + x^4 + x^3 + x^2 + x \\ \hline & x^4 + x^3 + x^2 + x + 1 \\ & x^4 + x^3 + x^2 + x + 1 \\ \hline & 0 \end{array}$$

Оскільки $x^5 + 1 = (x + 1)(x^4 + x^3 + x^2 + x + 1)$, то зрозуміло, що многочлен $x^5 + 1$ ділиться на $x^4 + x^3 + x^2 + x + 1$ без залишку, тому 5 є періодом. Оскільки період простого многочлена вибирається найменшим із чисел 15 та 5, то — це 5.

Таким чином, періоди трьох простих многочленів, які утворюють складний многочлен коду БЧХ (15, 5), відповідно становлять:

$$x^2 + x + 1 \rightarrow \text{період } 3,$$

$$x^4 + x + 1 \rightarrow \text{період } 15,$$

$$x^4 + x^3 + x^2 + x + 1 \rightarrow \text{період } 5.$$

Період складного многочлена, що складається з добутку трьох простих многочленів, дорівнює найбільшому спільному кратному періодів цих простих многочленів (LCM), а саме:

$$\textit{період складного многочлена} =$$

$$= \text{LCM}\left\{\textit{період}\left(x^2 + x + 1\right), \textit{період}\left(x^4 + x + 1\right), \textit{період}\left(x^4 + x^3 + x^2 + x + 1\right)\right\} =$$

$$= \text{LCM}\{3, 15, 5\} = 15$$

Період складного многочлена n з подвоєнням числа простих многочленів становить $x^n = 1$, а степінь подвійних многочленів не відповідає наведеній вище закономірності n. Тобто період складного многочлена n є степенем $n - 1$ многочлена коду $v(x)$, утвореного від цього складного многочлена:

$$v(x) = b_{n-1}x^{n-1} + b_{n-2}x^{n-2} + b_{n-3}x^{n-3} + \ldots + b_1x + b_0.$$

Спосіб створення циклічного коду

Отже, запишемо послідовність бітів циклічного коду, утвореного від складного многочлена з періодом n,

$$b_{n-1}x,\ b_{n-2},\ b_{n-3},\ \ldots,\ b_1,\ b_0.$$

Циклічний код — це код, утворений від складного многочлена, тому складний многочлен визначається так:

- період n складного многочлена — число бітів коду n;
- степінь m складного многочлена — число контрольних бітів m;
- число інформаційних бітів $k = n - m$.

При створенні коду БЧХ для формування складного многочлена, основа якого — послідовність основ α вихідного многочлена, які підносяться до степеня, вважається, що період складного многочлена збігається з періодом основи α.

Цілком природним є перехід до циклічного коду, оскільки він також утворений складним многочленом, побудованим на добутку довільних простих многочленів. Порядок утворення циклічного коду наведено на рис. 46.

Наприклад, із трьох простих многочленів, які повністю не збігаються:

$$P_1(x)\ (\textit{степінь}\ m_1, \textit{період}\ n_1),$$
$$P_2(x)\ (\textit{степінь}\ m_2, \textit{період}\ n_2),$$
$$P_3(x)\ (\textit{степінь}\ m_3, \textit{період}\ n_3),$$

створимо складний многочлен $G(x)$: $G(x) = P_1(x) \cdot P_2(x) \cdot P_3(x)$.

Період $G(x)$ – n= LCM {період $P_1(x)$, період $P_2(x)$, період $P_3(x)$} = LCM {n_1, n_2, n_3}; степінь — $m = m_1 + m_2 + m_3$. Циклічний код, утворений таким складним многочленом, має наступні параметри:

- число бітів коду n — період $G(x)$;
- число контрольних бітів m — степінь $G(x)$;
- число інформаційних бітів — $k = n - m$.

① Побудова складного многочлена $G(x)$ з добутку простих многочленів $P(x)$ — $G(x) = P_1(x) \cdot P_2(x) \cdot \ldots \cdot P_i(x)$

② Визначення періоду складного многочлена — Період $G(x)$ = LCM {період $P_1(x)$, період $P_2(x)$, ..., період $P_i(x)$}

③ Визначення числа інформаційних бітів — Період $G(x)$ = число бітів коду n; Степінь $G(x)$ = число контрольних бітів m; Число інформаційних бітів $k = n - m$

④ Визначення мінімальних кодових відстаней d за структурою контрольних бітів

⑤ Визначення корегувальної здатності t за мінімальним значенням кодової відстані d — $d = 2t + 1$ або $d = 2t + 2$

↓

Циклічний код (n, k) з корегувальною здатністю t

Рис. 46. Порядок утворення циклічного коду

Що стосується можливості виправлення помилок цим (n, k) циклічним кодом, то, як уже зазначалося, необхідно знайти співвідношення між інформаційними і контрольними бітами.

Наведений вище порядок утворення циклічного коду є випадком, коли прості многочлени, які утворюють складний многочлен, різні. У разі, коли прості многочлени однакові, складний поліном, складений із них, вважають єдиним, і його період необхідно визначити.

Розглянемо процес створення циклічного коду з різних простих многочленів, але перед цим для зручності визначимо молодший многочлен і його період.

19.2. ПРОСТИЙ МНОГОЧЛЕН І ЙОГО ПЕРІОД

Простий многочлен, як зазначалося вище, відповідає цілим простим числам, тому його не можна розкласти на множники. Многочлен, що розкладається на співмножники, ще називають многочленом, що розкладається; він відповідає числу складеному з цілих чисел.

Запишемо прості многочлени до шостого степеня включно.

- ***Простий многочлен першого степеня***

$$x,$$
$$x + 1.$$

- ***Простий многочлен другого степеня***

$$x^2 = x \cdot x,$$
$$x^2 + 1 = (x + 1)^2,$$
$$x^2 + x = x(x + 1),$$
$$x^2 + x + 1.$$

- ***Простий многочлен третього степеня***

$$x^3 = x \cdot x \cdot x,$$
$$x^3 + 1 = (x + 1)(x^2 + x + 1),$$
$$x^3 + x = x(x^2 + 1) = x(x + 1)^2,$$
$$x^3 + x^2 = x^2(x + 1),$$
$$x^3 + x + 1 \text{ (простий)},$$
$$x^3 + x^2 + x = x(x^2 + x + 1),$$
$$x^3 + x^2 + x + 1 = (x + 1)(x^2 + 1) = (x + 1)^3.$$

- ***Простий многочлен четвертого степеня***

$$x^4 = x \cdot x \cdot x \cdot x,$$
$$x^4 + 1 = (x^2 + 1)^2 = (x + 1)^4,$$
$$x^4 + x = x(x^3 + 1) = x(x + 1)(x^2 + x + 1),$$
$$x^4 + x^2 = x^2(x^2 + 1) = x^2(x + 1)^2,$$
$$x^4 + x^3 = x^3(x + 1),$$

$$x^4 + x + 1 \text{ (простий)},$$

$$x^4 + x^2 + 1 = (x^2 + x + 1)^2,$$

$$x^4 + x^3 + 1 \text{ (простий)},$$

$$x^4 + x^2 + x + 1 = (x + 1)(x^3 + x^2 + 1),$$

$$x^4 + x^3 + x + 1 = (x + 1)(x^3 + 1) = (x + 1)^2(x^2 + x + 1),$$

$$x^4 + x^3 + x^2 + 1 = (x + 1)(x^3 + x + 1),$$

$$x^4 + x^3 + x^2 + x + 1 \text{ (простий)}.$$

Тепер для визначення простого многочлена перевіримо їх по одному. Простий многочлен повинен мати вигляд:

$$x^n + \ldots + 1.$$

У цьому разі визначаючи прості многочлени, краще із самого початку виключити з розгляду многочлени іншого вигляду:

- ***Простий многочлен п'ятого степеня***

$$x^5 + 1 = (x + 1)(x^4 + x^3 + x^2 + x + 1),$$

$$x^5 + x + 1 = (x^2 + x + 1)(x^3 + x^2 + 1),$$

$$x^5 + x^2 + 1 \text{ (простий)},$$

$$x^5 + x^3 + 1 \text{ (простий)},$$

$$x^5 + x^4 + 1 = (x^2 + x + 1)(x^3 + x + 1),$$

$$x^5 + x^2 + x + 1 = (x + 1)(x^4 + x^3 + x^2 + 1) = (x + 1)^2(x^3 + x + 1),$$

$$x^5 + x^3 + x + 1 = (x + 1)(x^4 + x^3 + 1),$$

$$x^5 + x^4 + x + 1 = (x + 1)(x^4 + 1) = (x + 1)^5,$$

$$x^5 + x^3 + x^2 + 1 = (x + 1)(x^4 + x^3 + x + 1) = (x + 1)^3(x^2 + x + 1),$$

$$x^5 + x^4 + x^2 + 1 = (x + 1)(x^4 + x + 1),$$

$$x^5 + x^4 + x^3 + 1 = (x + 1)(x^4 + x^2 + x + 1) = (x + 1)^2(x^3 + x^2 + 1),$$

$$x^5 + x^3 + x^2 + x + 1 \text{ (простий)},$$

$$x^5 + x^4 + x^2 + x + 1 \text{ (простий)},$$

$$x^5 + x^4 + x^3 + x + 1 \text{ (простий)},$$

$$x^5 + x^4 + x^3 + x^2 + 1 \text{ (простий)}.$$

Виникає питання, чи є останній розглянутий многочлен простим многочленом, чи ні. Це можна з'ясувати, поділивши його на простий многочлен степеня, меншого ніж степінь даного многочлена. Проте перевірку може значно спростити, якщо виходити з того, парна чи непарна кількість членів даного многочлена.

Розглянемо випадок, коли кількість членів многочлена парна. Тоді обов'язково повинен бути присутній множник $x + 1$. Наприклад, для много-

члена $x^5 + 1$ запишемо

$$x^5 + 1 = x^5 + \underbrace{x^4 + x^4}_{0} + \underbrace{x^3 + x^3}_{0} + \underbrace{x^2 + x^2}_{0} + \underbrace{x + x}_{0} + 1 =$$

$$= x^4(x+1) + x^3(x+1) + x^2(x+1) + x(x+1) + (x+1) =$$

$$= (x+1)(x^4 + x^3 + x^2 + x + 1),$$

тоді доповнюючи цей вираз членами із степенями, яких у ньому немає, вдається отримати вираз $x + 1$ як співмножник.

У загальному випадку виділення множника $x + 1$ у разі парної кількості членів многочлена відбувається наступним чином.

Нехай у многочлені $F_n(x)$ із поля $GF(2)$ число членів x парне, тобто при

$$F_n(x) = \underbrace{x^n + x^{n-i} + \ldots + x^{n-k} + 1}_{парне}$$

$$F_n(1) = \underbrace{1 + 1 + \ldots + 1 + 1}_{непарне}$$

"1" є єдиною основою $F(x)$. Отже, $F(x)$ розкладається на множники так:

$$F_n(x) = (x+1)F_{n-1}(x).$$

Таким чином, як показано вище, у разі парного числа членів многочлена, виходячи з виділення $(x + 1)$ як співмножника при визначенні простого многочлена, про виключення многочленів, що мають парне число членів, не може бути й мови.

- ***Простий многочлен шостого степеня***

$$x^6 + x + 1 \text{ (простий)},$$

$$x^6 + x^2 + 1 = (x^3 + x + 1)^2,$$

$$x^6 + x^3 + 1 \text{ (простий)},$$

$$x^6 + x^4 + 1 = (x^3 + x^2 + 1)^2,$$

$$x^6 + x^5 + 1 \text{ (простий)},$$

$$x^6 + x^3 + x^2 + x + 1 = (x^2 + x + 1)(x^4 + x^3 + 1),$$

$$x^6 + x^4 + x^2 + x + 1 \text{ (простий)},$$

$$x^6 + x^5 + x^2 + x + 1 \text{ (простий)},$$

$$x^6 + x^4 + x^3 + x + 1 \text{ (простий)},$$

$$x^6 + x^5 + x^3 + x + 1 = (x^2 + x + 1)(x^4 + x^2 + 1) = (x^2 + x + 1)^3,$$

$$x^6 + x^5 + x^4 + x + 1 \text{ (простий)},$$

$$x^6 + x^4 + x^3 + x^2 + 1 = (x^2 + x + 1)(x^4 + x^3 + x^2 + 1),$$

$$x^6 + x^5 + x^3 + x^2 + 1 \text{ (простий)},$$

$$x^6 + x^5 + x^4 + x^2 + 1 \text{ (простий)},$$

$$x^6 + x^5 + x^4 + x^3 + 1 = (x^2 + x + 1)(x^4 + x + 1),$$

$$x^6 + x^5 + x^4 + x^3 + x^2 + x + 1 = (x^3 + x + 1)(x^3 + x^2 + 1).$$

З простими многочленами вище ніж сьомого степеня чинять аналогічно.

Далі знайдемо періоди простих многочленів. Період простого многочлена — це найменше значення n многочлена $x^n + 1$, що ділиться без залишку на простий многочлен.

Період простого многочлена першого степеня. Період простого многочлена першого степеня дорівнює 1: $2^1 - 1 = 1$. Отже, простим многочленом першого степеня, на який без залишку ділиться многочлен $x + 1$, є тільки многочлен $x + 1$. Тобто, періоди простих многочленів першого степеня мають вигляд: x — без періоду; $x + 1$ — період 1.

Період простого многочлена другого степеня. Період простого многочлена другого степеня дорівнює 3: $2^2 - 1 = 3$. Період 3 є простим числом і на множники не розкладається. Отже, період простого многочлена другого степеня — 3: $x^2 + x + 1$ має період 3.

Період простого многочлена третього степеня. Період простого многочлена третього степеня дорівнює 7: $2^3 - 1 = 7$. Період 7 також є простим числом і на множники не розкладається. Отже, період простих множників третього степеня — 7: $x^3 + x + 1$ має період 7, $x^3 + x^2 + 1$ має період 7.

Період простого многочлена четвертого степеня. Період простого многочлена четвертого степеня дорівнює 15: $2^4 - 1 = 15$. Період 15 можна розкласти на множники: $15 = 5 \cdot 3$. Серед співмножників число 5 більше, ніж степінь 4. Тоді як період простого многочлена можна розглядати два числа: 15 і 5. Отже, необхідно перевірити, чи ділиться $x^5 + 1$ на простий многочлен без залишку. Якщо ділиться без залишку, то період дорівнює 5, якщо не ділиться — 15.

Простих многочленів четвертого степеня є три: $x^4 + x + 1$ має період 15, $x^4 + x^2 + 1$ має період 15, $x^4 + x^3 + x^2 + x + 1$ має період 5.

Період простого многочлена п'ятого степеня. Період простого многочлена п'ятого степеня дорівнює 31: $2^5 - 1 = 31$. Період 31 є простим числом: на множники не розкладається. Отже, період простого многочлена п'ятого степеня дорівнює 31, тобто многочлени

$$x^5 + x^2 + 1,$$
$$x^5 + x^3 + 1,$$
$$x^5 + x^3 + x^2 + x + 1,$$
$$x^5 + x^4 + x^3 + x^2 + x + 1,$$
$$x^5 + x^4 + x^3 + x + 1,$$
$$x^5 + x^4 + x^3 + x^2 + 1$$

мають один і той самий період 31.

Період простого многочлена шостого степеня. Період простого многочлена шостого степеня дорівнює 63: $2^6 - 1 = 63$. Період 63 можна розкласти на такі множники: $63 = 7 \cdot 3 \cdot 3$. Отже, серед дільників 63 є чотири дільники, більших ніж степінь 6 простого многочлена: 63, 21, 9, 7. Серед них 7 — період простого многочлена третього степеня, тому його можна не розглядати як період многочлена шостого степеня.

Є дев'ять многочленів шостого степеня:

$$x^6 + x + 1,$$
$$x^6 + x^3 + 1,$$
$$x^6 + x^5 + 1,$$
$$x^6 + x^4 + x^2 + x + 1,$$
$$x^6 + x^5 + x^2 + x + 1,$$
$$x^6 + x^4 + x^3 + x + 1,$$
$$x^6 + x^5 + x^4 + x + 1,$$
$$x^6 + x^5 + x^3 + x^2 + 1,$$
$$x^6 + x^5 + x^4 + x^2 + 1.$$

Період цих простих многочленів дорівнює будь-якому з чисел: 63, 21, 9.

Якщо многочлен $x^9 + 1$ без залишку ділиться на простий многочлен, то період дорівнює 9, якщо $x^{21} + 1$ без залишку ділиться на простий многочлен, то — 21, і, нарешті, якщо ніякий з них не ділиться без залишку, то — 63. Поділимо $x^9 + 1$ і $x^{21} + 1$ на простий многочлен, при цьому у разі ділення без залишку простий многочлен можна виділити з початкового многочлена як множник. Таким чином, вирази $x^9 + 1$ і $x^{21} + 1$ слід розкласти на співмножники.

Періодом $x^9 + 1$ є число 9, його співмножником — 3. Отже, $x^9 + 1$ можна поділити без залишку на $x^3 + 1$, тобто розкласти на прості множники:

$$x^9 + 1 = (x^3 + 1)(x^6 + x^3 + 1) = (x + 1)(x^2 + x + 1)(x^6 + x^3 + 1). \qquad (33)$$

Серед співмножників $x^9 + 1$ є $x^6 + x^3 + 1$, тому $x^6 + x^3 + 1$ має період 9.

Періодом многочлена $x^{21} + 1$ є число 21, яке розкладається на множники: $21 = 7 \cdot 3$. Отже, $x^{21} + 1$ можна без залишку поділити на $x^7 + 1$ і $x^3 + 1$. Наприклад, виконаємо ділення на $x^7 + 1$:

$$\begin{array}{r|l} & x^{14} + x^7 + 1 \\ \hline x^7 + 1 & x^{21} \qquad\qquad + 1 \\ & \underline{x^{21} + x^{14}} \\ & \qquad x^{14} \quad + 1 \\ & \qquad \underline{x^{14} + x^7} \\ & \qquad\qquad x^7 + 1 \\ & \qquad\qquad \underline{x^7 + 1} \\ & \qquad\qquad\quad 0 \end{array}$$

Отримаємо $x^{21} + 1 = (x^7 + 1)(x^{14} + x^7 + 1)$. Крім того, $x^{21} + 1$ можна також поділити без залишку на $x^3 + 1$. Кожний з виразів $x^3 + 1$ і $x^7 + 1$ можна розкласти на множники таким чином:

$$x^3 + 1 = (x + 1)(x^2 + x + 1), \qquad (34)$$
$$x^7 + 1 = (x + 1)(x^3 + x + 1)(x^3 + x^2 + 1). \qquad (35)$$

Порівнюючи співмножники виразів $x^7 + 1$ і $x^3 + 1$, неважко з'ясувати, що вираз $x^2 + x + 1$ не є співмножником многочлена $x^7 + 1$. Отже, $x^{14} + x^7 + 1$ ділиться без залишку на $x^2 + x + 1$. Тоді, виходячи з результатів розрахунків, наведених на рис. 47, отримуємо

$$x^{14} + x^7 + 1 = (x^2 + x + 1)(x^{12} + x^{11} + x^9 + x^8 + x^6 + x^4 + x^3 + x + 1). \qquad (36)$$

Тепер розкладемо на множники многочлен $x^{12} + x^{11} + x^9 + x^8 + x^6 + x^4 + x^3 + x + 1$. Напевно, цей многочлен можна розкласти на два прості многочлени шостого степеня. В процесі розкладання $x^{21} + 1$ на співмножники з'являються такі вирази:

- простий многочлен першого степеня $x + 1$,
- простий многочлен другого степеня $x^2 + x + 1$,
- прості многочлени третього степеня: $x^3 + x + 1$, $x^3 + x^2 + 1$,

тому многочлен 12 степеня можна розкласти на добуток трьох простих многочленів четвертого степеня або двох простих многочленів шостого степеня.

$$
\begin{array}{r|l}
x^{14} + x^7 + 1 & x^2 + x + 1 \\
\hline
x^{14} + x^{13} + x^{12} & x^{12} + x^{11} + x^9 + x^8 + x^6 + x^4 + x^3 + x + 1 \\
\hline
x^{13} + x^{12} + x^7 + 1 & \\
x^{13} + x^{12} + x^{11} & \\
\hline
x^{11} + x^7 + 1 & \\
x^{11} + x^{10} + x^9 & \\
\hline
x^{10} + x^9 + x^7 + 1 & \\
x^{10} + x^9 + x^8 & \\
\hline
x^8 + x^7 + 1 & \\
x^8 + x^7 + x^6 & \\
\hline
x^6 + 1 & \\
x^6 + x^5 + x^4 & \\
\hline
x^5 + x^4 + 1 & \\
x^5 + x^4 + x^3 & \\
\hline
x^3 + 1 & \\
x^3 + x^2 + x & \\
\hline
x^2 + x + 1 & \\
x^2 + x + 1 & \\
\hline
0 &
\end{array}
$$

Рис. 47. Схема ділення $x^{14} + x^7 + 1$ на $x^2 + x + 1$

Проте, як показано вище, прості многочлени четвертого степеня є множниками многочлена $x^{15} + 1$, і, оскільки число 15 — період многочлена $x^{15} + 1$, не є дільником числа 21 — період многочлена $x^{21} + 1$, то і самого многочлена четвертого степеня немає серед співмножників $x^{21} + 1$.

Метод дослідження двох простих многочленів шостого степеня полягає в розгляді можливості ділення на всі шість простих поліномів. Виконаємо таке ділення за деякого наближення. Члени нижчих степенів — це:

$$... + x^3 + x + 1.$$

Наприклад, як члени нижчих степенів простого многочлена шостого степеня приймаються такі:

$$... + x + 1,$$
$$... + x^2 + 1.$$

Перемноживши їх:

$$(... + x + 1)(... + x^2 + 1) = ... + x^3 + x^2 + x + 1,$$

одержимо зайвий член — x^2. Цей член x^2 необхідно виключити, тому як члени нижчих степенів простого многочлена шостого степеня розглядатимемо такі члени:

$$... + x^2 + x + 1,$$
$$... + x^2 + 1.$$

Виконавши множення цих членів:

$$(... + x^2 + x + 1)(... + x^2 + 1) = ... + x^4 + x^3 + x + 1,$$

отримаємо необхідний вигляд членів нижчих степенів.

Прості многочлени шостого степеня, що мають члени нижчих степенів типу $... + x^2 + x + 1$, можна подати у вигляді

$$x^6 + x^4 + x^2 + x + 1,$$
$$x^6 + x^5 + x^2 + x + 1.$$

Поділимо многочлен степеня 12 із формули (36) на простий многочлен шостого степеня типу $... + x^2 + x + 1$ (рис. 48).

Неважко побачити, що ділення без залишку на простий многочлен типу $x^6 + x^4 + x^2 + x + 1$ можливе. Отже, простий многочлен можна розкласти на множники. Періодом двох простих многочленів шостого степеня, що містяться в поліномі $x^{21} + 1$, є число 21, тобто

$$x^6 + x^4 + x^2 + x + 1 \text{ має період } 21,$$
$$x^6 + x^5 + x^4 + x^2 + 1 \text{ має період } 21.$$

У простих многочленах шостого степеня, наведених вище, один поліном має період 9, а два — період 21. Отже, період решти шести простих многочленів дорівнює 63, а саме:

$$x^6 + x + 1,$$
$$x^6 + x^5 + 1,$$
$$x^6 + x^5 + x^2 + x + 1,$$
$$x^6 + x^4 + x^3 + x + 1,$$

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad x^6 \qquad + x^3 + x^2 \\
x^6 + x^5 + x^2 + x + 1)\ x^{12} + x^{11} \qquad + x^9 + x^8 \qquad + x^6 \qquad + x^4 + x^3 \qquad + x + 1 \\
\underline{\quad x^{12} + x^{11} \qquad\qquad x^8 + x^7 + x^6 \qquad\qquad\qquad\qquad\qquad\qquad} \\
\qquad\qquad\qquad\quad x^9 \qquad + x^7 \qquad\qquad + x^4 + x^3 \qquad + x + 1 \\
\qquad\qquad\qquad\quad \underline{x^9 + x^8 \qquad\qquad + x^5 + x^4 + x^3 \qquad\qquad\qquad} \\
\qquad\qquad\qquad\qquad\quad x^8 + x^7 \qquad x^5 \qquad\qquad\qquad + x + 1 \\
\qquad\qquad\qquad\qquad\quad \underline{x^8 + x \qquad\qquad + x^4 + x^3 + x^2 \qquad\qquad} \\
\qquad\qquad\qquad\qquad\qquad\qquad\quad x^5 + x^4 + x^3 + x^2 + x + 1
\end{array}
$$

Не ділиться без залишку

$$
\begin{array}{l}
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad x^6 + x^5 + x^4 \qquad + x^2 \qquad + 1 \\
x^6 + x^4 + x^2 + x + 1)\ x^{12} + x^{11} \qquad + x^9 + x^8 \qquad + x^6 \qquad + x^4 + x^3 \qquad + x + 1 \\
\underline{\quad x^{12} \qquad + x^{10} \qquad x^8 + x^7 + x^6 \qquad\qquad\qquad\qquad\qquad\qquad} \\
\qquad\qquad\quad x^{11} + x^{10} + x^9 \qquad + x^7 \qquad\qquad + x^4 + x^3 \qquad + x + 1 \\
\qquad\qquad\quad \underline{x^{11} \qquad + x^9 \qquad + x^7 + x^6 + x^5 \qquad\qquad\qquad\qquad} \\
\qquad\qquad\qquad\quad x^{10} \qquad\qquad + x^6 + x^5 + x^4 + x^3 \qquad + x + 1 \\
\qquad\qquad\qquad\quad \underline{x^{10} \qquad + x^8 \qquad + x^6 + x^5 + x^4 \qquad\qquad\qquad} \\
\qquad\qquad\qquad\qquad\quad x^8 \qquad\qquad\qquad + x^3 \qquad + x + 1 \\
\qquad\qquad\qquad\qquad\quad \underline{x^8 \qquad + x^6 \qquad + x^4 + x^3 + x^2 \qquad\qquad} \\
\qquad\qquad\qquad\qquad\qquad\quad x^6 \qquad + x^4 \qquad + x^2 + x + 1 \\
\qquad\qquad\qquad\qquad\qquad\quad \underline{x^6 \qquad + x^4 \qquad + x^2 + x + 1} \\
\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad 0
\end{array}
$$

Ділиться без залишку

Рис. 48. Схема ділення многочлена $x^{12} + x^{11} + x^9 + x^8 + x^6 + x^4 + x^3 + x + 1$ на простий многочлен шостого степеня

$$x^6 + x^5 + x^4 + x + 1,$$
$$x^6 + x^5 + x^3 + x^2 + 1.$$

Періоди простих многочленів, що мають степінь від 1 до 6 включно, наведено в табл. 27.

Тепер розглянемо також періоди складних многочленів циклічного коду, що складається з декількох однакових простих многочленів. Наприклад, знайдемо період для складного многочлена, утвореного з многочленів третього степеня $x^3 + x + 1$. Період такого простого многочлена — $x^3 + x + 1$ — дорівнює 7. Цей факт має той сенс, що на нього без залишку ділиться многочлен $x^7 + 1$, тобто

$$x^7 + 1 = Q(x)(x^3 + x + 1),$$

тому $x^3 + x + 1$ є співмножником $x^7 + 1$.

Таблиця 27. Періоди простих многочленів від шостого степеня і нижче

Степінь	Простий многочлен	Період
1	x	—
	$x+1$	1
2	x^2+x+1	3
3	x^3+x+1	7
	x^3+x^2+1	7
4	x^4+x+1	15
	x^4+x^3+1	15
	$x^4+x^3+x^2+x+1$	5
5	x^5+x^2+1	31
	x^5+x^3+1	31
	$x^5+x^3+x^2+x+1$	31
	$x^5+x^4+x^2+x+1$	31
	$x^5+x^4+x^3+x+1$	31
	$x^5+x^4+x^3+x^2+1$	31
6	x^6+x+1	63
	x^6+x^3+1	9
	x^6+x^5+1	63
	$x^6+x^4+x^2+x+1$	21
	$x^6+x^5+x^2+x+1$	63
	$x^6+x^4+x^3+x+1$	63
	$x^6+x^5+x^4+x+1$	63
	$x^6+x^5+x^3+x^2+1$	63
	$x^6+x^5+x^4+x^2+1$	21

Якщо $x^3 + x + 1$ піднести до квадрата:

$$(x^7 + 1)^2 = Q^2(x)(x^3 + x + 1)^2,$$

то отримаємо многочлен

$$(x^7 + 1)^2 = x^{14} + 1.$$

Відповідно до якого період многочлена $(x^3 + x + 1)^2$ дорівнює 14.

Далі знайдемо період многочлена, піднесеного до кубу:

$$(x^7 + 1)^3 = Q^3(x)(x^3 + x + 1)^3.$$

Однак многочлен $x^7 + 1$ до вигляду $x^n + 1$ не зводиться. Відомо, проте, що при піднесенні до квадрата виразу $x^3 + x + 1$ період дорівнював 14, тобто

$$(x^7 + 1)^2 = x^{14} + 1 = Q^2(x)(x^3 + x + 1)^2.$$

Тут слід звернути увагу на співвідношення

$$(x^n + 1)^2 = x^{2n} + 1.$$

Згідно з законом $(x^{2n} + 1)^2 = x^{4n} + 1$, $(x^{4n} + 1)^2 = x^{8n} + 1$, ..., для подання многочлена у формі $x^n + 1$ у разі піднесення його до степеня необхідно багаторазово виконувати операції піднесення до степеня числа 2: $2(2^1)$, $4\ (2^2)$, $8\ (2^3)$.

Отже, коли многочлен складається з виразів $x^3 + x + 1$, то

$$(x^7 + 1)^1 = x^7 + 1 = Q(x)(x^3 + x + 1),$$
$$(x^7 + 1)^2 = x^{14} + 1 = Q^2(x)(x^3 + x + 1)^2,$$
$$(x^7 + 1)^4 = x^{28} + 1 = Q^4(x)(x^3 + x + 1)^4,$$
$$(x^7 + 1)^8 = x^{56} + 1 = Q^8(x)(x^3 + x + 1)^8,$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$

Це і є співвідношення між періодом n і степенем многочлена.

Відповідно, у разі піднесення $x^3 + x + 1$ до степеня 3, період — $(x^7 + 1)^4$, а саме:

$$x^{28} + 1 = Q^4(x)(x^3 + x + 1)^4 = Q^4(x)(x^3 + x + 1)(x^3 + x + 1)^3,$$

і саме в цій формі його необхідно визначати. Тому, остаточно $(x^3 + x + 1)^3$ має період 28.

Таким чином, у разі утворення складного многочлена з простих однотипних многочленів період знайти нескладно, виходячи з періодів простих многочленів.

На цьому підготовка по створенню циклічного коду закінчується. Проте перед остаточним формуванням циклічного коду слід розглянути співвідношення між простим і первинним многочленами.

19.3. НЕПРИМІТИВНИЙ КОД БЧХ

Раніше поняття коду БЧХ було сформульовано так: якщо α вважати базисом примітивного многочлена, то код створюється з використанням породжувального многочлена, базис якого — послідовність базисів, піднесених до степеня, тобто α, α^2, ..., α^{2t-1}, α^{2t}. Цей базис α є примітивним базисом поля $GF(2^M)$, що створюється з простого многочлена (у степені M). Іноді примітивний код БЧХ — це код, який утворюється з породжувального многочлена і містить як базис степеневу послідовність примітивного базису α. Використовуючи базис β, що не є примітивним базисом поля $GF(2^M)$, яке утворюється з простого многочлена (у степені M), можна утворити породжувальний многочлен, який містить як базис степеневу послідовність з цих базисів β.

Оскільки базис породжувального многочлена формується із степеневої послідовності базисів β код, що створюється з цього породжувального многочлена є кодом БЧХ. Проте, оскільки в цьому випадку базис не є примітивним, то і відповідний код БЧХ є непримітивним кодом.

Для того щоб можна було створити непримітивний код БЧХ, необхідно період простого многочлена $n = 2^M - 1$ подати у вигляді співмножників.

Проаналізуємо примітивний многочлен щодо степеня M і періоду, починаючи від четвертого степеня і вище.

Степінь простого многочлена	Період
$M = 4$	$n = 2^4 - 1 = 15 = 5 \cdot 3$
$M = 5$	$n = 2^5 - 1 = 31$
$M = 6$	$n = 2^6 - 1 = 63 = 7 \cdot 3 \cdot 3$
$M = 7$	$n = 2^7 - 1 = 127$
$M = 8$	$n = 2^8 - 1 = 255 = 51 \cdot 5$
...	...

Якщо використовувати прості многочлени 4-, 6-, 8-го порядків, то можна створити непримітивний код БЧХ. Наприклад, використовуючи простий многочлен 6-го порядку $x^6 + x + 1$, покажемо, як можна створити непримітивний код БЧХ. Базис α простого многочлена є простим базисом поля $GF(2^6)$. Отже, ним можна позначити всі базиси, за винятком нульового базису поля $GF(2^6)$. Цей простий базис α є базисом виразу $x^6 + x + 1$, тому $\alpha^6 + \alpha + 1 = 0$ і $\alpha^6 = \alpha + 1$. Одночасно з використанням такої форми запису, можна позначити простим базисом α всі базиси, крім нульового базису, поля $GF(2^6)$ у виразах від п'ятого степеня і нижче. У табл. 28 наведено многочлени з базисом α поля $GF(2^6)$.

Таблиця 28. Многочлени з базисом α і базис поля $GF(2^6)$

Базис поля $GF(2^6)$	Многочлен з базисом α	Базис поля $GF(2^6)$	Многочлен з базисом α
0	----------	α^{31}	$\alpha^5 + \alpha^2 + 1$
$\alpha^0 (= \alpha^{63})$	1	α^{32}	$\alpha^3 + 1$
α	α	α^{33}	$\alpha^4 + \alpha$
α^2	α^2	α^{34}	$\alpha^5 + \alpha^2$
α^3	α^3	α^{35}	$\alpha^3 + \alpha + 1$
α^4	α^4	α^{36}	$\alpha^4 + \alpha^2 + \alpha$
α^5	α^5	α^{37}	$\alpha^5 + \alpha^3 + \alpha^2$
α^6	$\alpha + 1$	α^{38}	$\alpha^4 + \alpha^3 + \alpha + 1$
α^7	$\alpha^2 + \alpha$	α^{39}	$\alpha^5 + \alpha^4 + \alpha^2 + \alpha$
α^8	$\alpha^3 + \alpha^2$	α^{40}	$\alpha^5 + \alpha^3 + \alpha^2 + \alpha + 1$
α^9	$\alpha^4 + \alpha^3$	α^{41}	$\alpha^4 + \alpha^3 + \alpha^2 + 1$
α^{10}	$\alpha^5 + \alpha^4$	α^{42}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha$
α^{11}	$\alpha^5 + \alpha + 1$	α^{43}	$\alpha^5 + \alpha^4 + \alpha^2 + \alpha + 1$
α^{12}	$\alpha^2 + 1$	α^{44}	$\alpha^5 + \alpha^3 + \alpha^2 + 1$
α^{13}	$\alpha^3 + \alpha$	α^{45}	$\alpha^4 + \alpha^3 + 1$
α^{14}	$\alpha^4 + \alpha^2$	α^{46}	$\alpha^5 + \alpha^4 + \alpha$
α^{15}	$\alpha^5 + \alpha^3$	α^{47}	$\alpha^5 + \alpha^2 + \alpha + 1$
α^{16}	$\alpha^4 + \alpha + 1$	α^{48}	$\alpha^3 + \alpha^2 + 1$
α^{17}	$\alpha^2 + \alpha$	α^{49}	$\alpha^4 + \alpha^3 + \alpha$
α^{18}	$\alpha^3 + \alpha^2 + \alpha + 1$	α^{50}	$\alpha^5 + \alpha^4 + \alpha^2$
α^{19}	$\alpha^4 + \alpha^3 + \alpha^2 + \alpha$	α^{51}	$\alpha^5 + \alpha^3 + \alpha + 1$
α^{20}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2$	α^{52}	$\alpha^4 + \alpha^2 + 1$
α^{21}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha + 1$	α^{53}	$\alpha^5 + \alpha^3 + \alpha$
α^{22}	$\alpha^5 + \alpha^4 + \alpha^2 + 1$	α^{54}	$\alpha^4 + \alpha^2 + \alpha + 1$
α^{23}	$\alpha^5 + \alpha^3 + 1$	α^{55}	$\alpha^5 + \alpha^3 + \alpha^2 + \alpha$
α^{24}	$\alpha^4 + 1$	α^{56}	$\alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1$
α^{25}	$\alpha^5 + \alpha$	α^{57}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha$
α^{26}	$\alpha^2 + \alpha + 1$	α^{58}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1$
α^{27}	$\alpha^3 + \alpha^2 + \alpha$	α^{59}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + 1$
α^{28}	$\alpha^4 + \alpha^3 + \alpha^2$	α^{60}	$\alpha^5 + \alpha^4 + \alpha^3 + 1$
α^{29}	$\alpha^5 + \alpha^4 + \alpha^3$	α^{61}	$\alpha^5 + \alpha^4 + 1$
α^{30}	$\alpha^5 + \alpha^4 + \alpha + 1$	α^{62}	$\alpha^5 + 1$

Періодичність такого простого многочлена $2^6 - 1 = 63$. Цей період можна розкласти на множники: $63 = 7 \cdot 3 \cdot 3$.

Таким чином, розглядаючи простий многочлен шостого порядку, маємо, що базис поля $GF(2^6)$ включає в себе базиси, які мають періоди 9 і 21.

Визначаючи період многочлена шостого порядку, що не зводиться, слід звертати увагу на існування многочленів, які не зводяться, з періодами 21 і 9 (див. табл. 26). Базис поля $GF(2^6)$ містить базиси з такими періодами. Наприклад, якщо узяти базис α^7, то можна записати, що $(\alpha^7)^9 = \alpha^{63} = 1$. Отже, зрозуміло, що базис α^7 має період 9. Якщо проаналізувати періоди базисів β, де у виразі $\beta = \alpha^7$, α^7 — базис, то із результатів, наведених у табл. 29, випливає, що $\beta^9 = 1$, тобто період базису β дорівнює 9.

Тоді і період породжувального многочлена дорівнюватиме 21. У цьому випадку можна спробувати створити 21-бітовий непримітивний код БЧХ, використовуючи базис поля $GF(2^6)$.

Як бачимо з табл. 30, для базису β з періодом 21 ($\beta = \alpha^3$ (базис α^3)) можна записати $(\alpha^3)^{21} = \alpha^{63} = 1$ (α^3 — базис поля $GF(2^6)$).

Отже, період 21 базису β можна вважати базисом породжувального многочлена. Код, утворений з породжувального многочлена, який має як базис степеневу послідовність базисів $\beta(=\alpha^7)$ поля $GF(2^6)$, що складається із значень β, β^2, ..., є 21-бітовим кодом БЧХ SEC. Для того щоб створити такий код БЧХ, перш за все необхідно знайти найменший многочлен для основ β^2 і β. Окрім многочлена з базисом β, є також інші базиси, які утворюють ряд:

$$\beta, \beta^2, \beta^4, \beta^8, \beta^{16}, \beta^{32} = \beta, \ldots$$

Таблиця 29. Періодичність базису $\beta = \alpha^7$

Період базису	Базис поля $GF(2^6)$	Многочлен з базисом α
β^0	α^0	1
β	α^7	$\alpha^2 + \alpha$
β^2	α^{14}	$\alpha^4 + \alpha^2$
β^3	α^{21}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha + 1$
β^4	α^{28}	$\alpha^4 + \alpha^3 + \alpha^2$
β^5	α^{35}	$\alpha^3 + \alpha + 1$
β^6	α^{42}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha$
β^7	α^{49}	$\alpha^4 + \alpha^3 + \alpha$
β^8	α^{56}	$\alpha^4 + \alpha^3 + \alpha^2 + \alpha + 1$
$\beta^9(=\beta^0)$	$\alpha^{63}(=\alpha^0)$	1

Таблиця 30. Період базису $\beta = (\alpha^3)$

Період базису β	Базис поля $GF(2^6)$	Многочлен з базисом α
β^0	α^0	1
β^1	α^3	α^3
β^2	α^6	$\alpha + 1$
β^3	α^9	$\alpha^4 + \alpha^3$
β^4	α^{12}	$\alpha^2 + 1$
β^5	α^{15}	$\alpha^5 + \alpha^3$
β^6	α^{18}	$\alpha^3 + \alpha^2 + \alpha + 1$
β^7	α^{21}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha + 1$
β^8	α^{24}	$\alpha^4 + 1$
β^9	α^{27}	$\alpha^3 + \alpha^2 + \alpha$
β^{10}	α^{30}	$\alpha^5 + \alpha^4 + \alpha + 1$
β^{11}	α^{33}	$\alpha^4 + \alpha$
β^{12}	α^{36}	$\alpha^4 + \alpha^2 + \alpha$
β^{13}	α^{39}	$\alpha^5 + \alpha^4 + \alpha^2 + \alpha$
β^{14}	α^{42}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha$
β^{15}	α^{45}	$\alpha^4 + \alpha^3 + 1$
β^{16}	α^{48}	$\alpha^3 + \alpha^2 + 1$
β^{17}	α^{51}	$\alpha^5 + \alpha^3 + \alpha + 1$
β^{18}	α^{54}	$\alpha^4 + \alpha^2 + \alpha + 1$
β^{19}	α^{57}	$\alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha$
β^{20}	α^{60}	$\alpha^5 + \alpha^4 + \alpha^3 + 1$
β^{21} (= β^0)	α^{63}	1

Отже, базиси поля $GF(2^6)$ — ряд базисів типу β, β^2, β^4, β^8, β^{11}, β^{16} — містяться в одному і тому самому мінімальному многочлені $M_1(x)$. При цьому

$$M_1(x)=(x-\beta)(x-\beta^2)(x-\beta^4)(x-\beta^8)(x-\beta^{11})(x-\beta^{16})=x^6+x^4+x^2+x+1.$$

Процес обчислення многочлена $M_1(x)$ продемонстрований на рис. 49.

$$M_1(x)=(x+\beta)(x+\beta^2)(x+\beta^4)(x+\beta^8)(x+\beta^{11})(x+\beta^{16})=x^6+Ax^5+Bx^4+Cx^3+Dx^2+Ex+F$$

Обчислення коефіцієнта A біля x^5

$$A=\beta+\beta^2+\beta^4+\beta^8+\beta^{11}+\beta^{16}=(\alpha^3)+(\alpha+1)+(\alpha^2+1)+(\alpha^4+1)+(\alpha^4+\alpha)+(\alpha^3+\alpha^2+1)=0$$

Обчислення коефіцієнта B біля x^4

$$B=\beta\cdot\beta^2+\beta\cdot\beta^4+\beta\cdot\beta^8+\beta\cdot\beta^{11}+\beta\cdot\beta^{16}+\beta^2\cdot\beta^4+\beta^2\cdot\beta^8+\beta^2\cdot\beta^{11}+\beta^2\cdot\beta^{16}+\beta^4\cdot\beta^8+$$
$$+\beta^4\cdot\beta^{11}+\beta^4\cdot\beta^{16}+\beta^8\cdot\beta^{11}+\beta^8\cdot\beta^{16}+\beta^{11}\cdot\beta^{16}=\beta^5+\beta^9+\beta^{10}+\beta^{13}+\beta^{15}+\beta^{17}+\beta^{18}+$$
$$+\beta^{19}+\beta^{20}=(\alpha^5+\alpha^3)+(\alpha^3+\alpha^2+\alpha)+(\alpha^5+\alpha^4+\alpha+1)+(\alpha^5+\alpha^4+\alpha^2+\alpha)+(\alpha^4+\alpha^3+1)+$$
$$+(\alpha^5+\alpha^3+\alpha+1)+(\alpha^4+\alpha^2+\alpha+1)+(\alpha^5+\alpha^4+\alpha^3+\alpha^2+\alpha)+(\alpha^5+\alpha^4+\alpha^3+1)=1$$

Обчислення коефіцієнта C біля x^3

$$C=\beta\cdot\beta^2\cdot\beta^4+\beta\cdot\beta^2\cdot\beta^8+\beta\cdot\beta^2\cdot\beta^{11}+\beta\cdot\beta^2\cdot\beta^{16}+\beta\cdot\beta^4\cdot\beta^8+\beta\cdot\beta^4\cdot\beta^{11}+\beta\cdot\beta^4\cdot\beta^{16}+$$
$$+\beta\cdot\beta^8\cdot\beta^{11}+\beta\cdot\beta^8\cdot\beta^{16}+\beta\cdot\beta^{11}\cdot\beta^{16}+\beta^2\cdot\beta^4\cdot\beta^8+\beta^2\cdot\beta^4\cdot\beta^{11}+\beta^2\cdot\beta^4\cdot\beta^{16}+\beta^2\cdot\beta^8\cdot\beta^{11}+$$
$$+\beta^2\cdot\beta^8\cdot\beta^{16}+\beta^2\cdot\beta^{11}\cdot\beta^{16}+\beta^4\cdot\beta^8\cdot\beta^{11}+\beta^4\cdot\beta^8\cdot\beta^{16}+\beta^4\cdot\beta^{11}\cdot\beta^{16}+\beta^8\cdot\beta^{11}\cdot\beta^{16}=$$
$$=\beta+\beta^2+\beta^4+\beta^5+\beta^7+\beta^9+\beta^{10}+\beta^{11}+\beta^{13}+\beta^{14}+\beta^{16}+\beta^{17}+\beta^{19}+\beta^{20}=(\alpha^3)+(\alpha+1)+$$
$$+(\alpha^2+1)+(\alpha^5+\alpha^3)+(\alpha^5+\alpha^4+\alpha^3+\alpha+1)+(\alpha^3+\alpha^2+1)+(\alpha^5+\alpha^3+\alpha+1)+(\alpha^5+\alpha^4+\alpha^3+\alpha^2+\alpha)+$$
$$+(\alpha^5+\alpha^4+\alpha^3+1)=0$$

Обчислення коефіцієнта D біля x^2

$$D=\beta\cdot\beta^2\cdot\beta^8\cdot\beta^{11}+\beta\cdot\beta^2\cdot\beta^4\cdot\beta^{11}+\beta\cdot\beta^2\cdot\beta^4\cdot\beta^{16}+\beta\cdot\beta^2\cdot\beta^8\cdot\beta^{11}+\beta\cdot\beta^2\cdot\beta^8\cdot\beta^{16}+$$
$$+\beta\cdot\beta^2\cdot\beta^{11}\cdot\beta^{16}+\beta\cdot\beta^4\cdot\beta^8\cdot\beta^{11}+\beta\cdot\beta^4\cdot\beta^8\cdot\beta^{16}+\beta\cdot\beta^4\cdot\beta^{11}\cdot\beta^{16}+\beta\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}+$$
$$+\beta^2\cdot\beta^4\cdot\beta^8\cdot\beta^{11}+\beta^2\cdot\beta^4\cdot\beta^8\cdot\beta^{16}+\beta^2\cdot\beta^4\cdot\beta^{11}\cdot\beta^{16}+\beta^2\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}+\beta^4\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}=$$
$$=\beta+\beta^2+\beta^3+\beta^4+\beta^6+\beta^8+\beta^{11}+\beta^{12}+\beta^{16}=(\alpha^3)+(\alpha+1)+(\alpha^4+\alpha^3)+(\alpha^2+1)+$$
$$+(\alpha^3+\alpha^2+\alpha+1)+(\alpha^4+1)+(\alpha^4+\alpha)+(\alpha^4+\alpha^2+\alpha)+(\alpha^3+\alpha^2+1)=1$$

Обчислення коефіцієнта E біля x

$$E=\beta\cdot\beta^2\cdot\beta^4\cdot\beta^8\cdot\beta^{11}+\beta\cdot\beta^2\cdot\beta^4\cdot\beta^8\cdot\beta^{16}+\beta\cdot\beta^2\cdot\beta^4\cdot\beta^{11}\cdot\beta^{16}+\beta\cdot\beta^2\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}+$$
$$+\beta\cdot\beta^4\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}+\beta^2\cdot\beta^4\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}=\beta^5+\beta^{10}+\beta^{13}+\beta^{17}+\beta^{19}+\beta^{20}=$$
$$=(\alpha^5+\alpha^3)+(\alpha^5+\alpha^4+\alpha+1)+(\alpha^5+\alpha^4+\alpha^2+\alpha)+(\alpha^5+\alpha^3+\alpha+1)+(\alpha^5+\alpha^4+\alpha^3+\alpha^2+\alpha)+$$
$$+(\alpha^5+\alpha^4+\alpha^3+1)=1$$

Обчислення F

$$F=\beta\cdot\beta^2\cdot\beta^4\cdot\beta^8\cdot\beta^{11}\cdot\beta^{16}=\beta^{42}=1$$

$$\downarrow$$

$$M_1(x)=x^6+x^4+x^2+x+1$$

Рис. 49. Схема послідовності обчислення $M_1(x)$

Таким чином, породжувальний многочлен —

$$G(x) = \text{LCM}\ \{\textit{мінімальний многочлен базису}\ \beta,\ \textit{мінімальний многочлен базису}\ \beta^2\} = \beta^2\ \text{LCM}\{M_1(x),\ M_1(x)\} = M_1(x).$$

Визначаючи період незвідного многочлена, необхідно досліджувати два наявні 21-періодні незвідні многочлени, використовуючи незвідний многочлен шостого порядку. Один з таких многочленів — це $M_1(x)$. Визначимо період базису β породжувального многочлена. Оскільки

$$\beta^6 + \beta^4 + \beta^2 + \beta + 1 = 0, \tag{37}$$

то

$$\beta^6 = \beta^4 + \beta^2 + \beta + 1.$$

Звідси неважко знайти степеневий вираз базису β:

$$\begin{aligned}
\beta^0 &= 1\\
\beta^1 &= \beta\\
\beta^2 &= \beta^2\\
\beta^3 &= \beta^3\\
\beta^4 &= \beta^4\\
\beta^5 &= \beta^5\\
\beta^6 &= \beta^4 + \beta^2 + \beta + 1\\
\beta^7 &= \beta^5 + \beta^3 + \beta^2 + \beta\\
\beta^8 &= \beta^3 + \beta + 1\\
\beta^9 &= \beta^4 + \beta^2 + \beta\\
\beta^{10} &= \beta^5 + \beta^3 + \beta^2\\
\beta^{11} &= \beta^3 + \beta^2 + \beta + 1\\
\beta^{12} &= \beta^4 + \beta^3 + \beta^2 + \beta\\
\beta^{13} &= \beta^5 + \beta^4 + \beta^3 + \beta^2\\
\beta^{14} &= \beta^5 + \beta^3 + \beta^2 + \beta + 1\\
\beta^{15} &= \beta^3 + 1\\
\beta^{16} &= \beta^4 + \beta\\
\beta^{17} &= \beta^5 + \beta^2\\
\beta^{18} &= \beta^4 + \beta^3 + \beta^2 + \beta + 1\\
\beta^{19} &= \beta^5 + \beta^4 + \beta^3 + \beta^2 + \beta\\
\beta^{20} &= \beta^5 + \beta^3 + \beta + 1\\
\beta^{21}\ (= \beta^0) &= 1
\end{aligned} \tag{38}$$

Рівності (38) — вдалий розподіл співвідношення інформаційних і перевірних бітів. У такому непримітивному коді БЧХ, період породжувального многочлена дорівнює 21, степінь — 6, тому — це є код БЧХ (21, 15) SEC. На рис. 50 наведено структуру цього коду.

Інформаційні біти															Контрольні біти					
a_{14}	a_{13}	a_{12}	a_{11}	a_{10}	a_9	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	c_5	c_4	c_3	c_2	c_1	c_0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

$$c_5 = a_{14} + a_{13} + a_{11} + a_8 + a_7 + a_4 + a_1 \pmod 2$$
$$c_4 = a_{13} + a_{12} + a_{10} + a_7 + a_6 + a_3 + a_0 \pmod 2$$
$$c_3 = a_{14} + a_{13} + a_{12} + a_9 + a_8 + a_7 + a_6 + a_5 + a_4 + a_2 + a_1 \pmod 2$$
$$c_2 = a_{13} + a_{12} + a_{11} + a_8 + a_7 + a_6 + a_5 + a_4 + a_3 + a_1 + a_0 \pmod 2$$
$$c_1 = a_{14} + a_{13} + a_{12} + a_{10} + a_8 + a_6 + a_5 + a_3 + a_2 + a_1 + a_0 \pmod 2$$
$$c_0 = a_{14} + a_{12} + a_9 + a_8 + a_5 + a_2 + a_0 \pmod 2$$

Рис. 50. Структура непримітивного коду БЧХ (21, 15) SEC

Порівнюючи розглянутий вище циклічний код (21, 16), з урахуванням даних, поданих у табл. 31, і код БЧХ (21, 15) SEC, можна дійти висновку, що число інформаційних бітів циклічного коду більше на один біт. А саме: у коді БЧХ (21, 15) на одну перевірну конфігурацію менше, ніж у циклічному коді за однієї і тієї самої здатності до виправлення, тому циклічний код (21, 16) ефективніший.

Повернемося до непримітивного коду БЧХ, отриманого з породжувального многочлена, який містить як базис степеневу послідовність базисів β поля $GF(2^6)$: β, β^2, β^3, β^4. Як буде показано далі, цей код є також кодом БЧХ DEC.

Породжувальний многочлен неможливо утворити, якщо не визначити мінімальні многочлени для кожного базису. Проте для деяких базисів мінімальні многочлени уже визначені, а саме мінімальні многочлени для базисів

$$\beta,\ \beta^2,\ \beta^4 = M_1(x) = x^6 + x^4 + x + 1.$$

Тому, якщо знайти мінімальний многочлен для β^3, то можна утворити породжувальний многочлен.

У мінімальному многочлені з базисом β^3, окрім власне β^3, є також інші корені. Можна знайти ці корені з ряду β^3, β^6, β^{12}, $\beta^{24} = \beta^3$, ..., а саме — це три корені мінімального многочлена з основою β^3 : β^3, β^6, β^{12}.

За результатами обчислень, поданих на рис. 51, для $M_3(x)$ — мінімального многочлена, маємо, що

$$M_3(x) = (x - \beta^3)(x - \beta^6)(x - \beta^{12}) = x^3 + x^2 + 1.$$

Таблиця 31. Циклічні коди (21, 16) і БЧХ (21, 15)

Параметр	Непримітивний код БЧХ (21, 15)	Циклічний код (21, 16)
Загальне число бітів у коді	21	21
Число інформаційних бітів	15	16
Число перевірних бітів	6	5
Вигляд породжувального многочлена	$x^6 + x^4 + x^2 + x + 1$	$x^5 + x^4 + 1$
Здатність до виправлення помилок	SEC	SEC

$$M_3(x) = (x + \beta^3)(x + \beta^6)(x + \beta^{12}) = x^3 + Ax^2 + Bx + C$$

Обчислення коефіцієнта A біля x^2

$$A = \beta^3 + \beta^6 + \beta^{12} = (\alpha^4 + \alpha^3)(\alpha^3 + \alpha^2 + \alpha + 1) + (\alpha^4 + \alpha^2 + \alpha) = 1$$

Обчислення коефіцієнта B біля x

$$B = \beta^3 \cdot \beta^6 + \beta^3 \cdot \beta^{12} + \beta^6 \cdot \beta^{12} = \beta^9 + \beta^{15} + \beta^{19} =$$
$$= (\alpha^3 + \alpha^2 + \alpha) + (\alpha^4 + \alpha^3 + 1) + (\alpha^4 + \alpha^2 + \alpha + 1) = 0$$

Обчислення C

$$C = \beta^3 \cdot \beta^6 \cdot \beta^{12} = \beta^{21} = 1$$

$$\downarrow$$

$$M_3(x) = x^3 + x^2 + 1$$

Рис. 51. Схема обчислення мінімального многочлена $M_3(x)$

Мінімальні многочлени для базисів β, β^2, β^3, β^4 визначені, тому породжувальний многочлен можна записати так:

$$G(x) = \text{LCM}\{\textit{мінімальний многочлен базису } \beta, \textit{ мінімальний многочлен базису } \beta^2, \textit{ мінімальний многочлен базису } \beta^3, \textit{ мінімальний многочлен базису } \beta^4\} =$$
$$= \text{LCM}\{M_1(x), M_1(x), M_3(x), M_1(x)\} = M_1(x) \cdot M_3(x) = (x^6 + x^4 + x^2 + x + 1)(x^3 + x^2 + 1) =$$
$$= x^9 + x^8 + x^7 + x^5 + x^4 + x + 1.$$

Коренем породжувального многочлена $G(x)$ є базис β, тому маємо

$$\beta^9 + \beta^8 + \beta^7 + \beta^5 + \beta^4 + \beta + 1 = 0.$$

Використовуючи співвідношення $\beta^9 = \beta^8 + \beta^7 + \beta^5 + \beta^4 + \beta + 1$, можна знайти многочлени базису β і степені цього базису:

$$\begin{aligned}
\beta^0 &= 1\\
\beta^1 &= \beta\\
\beta^2 &= \beta^2\\
\beta^3 &= \beta^3\\
\beta^4 &= \beta^4\\
\beta^5 &= \beta^5\\
\beta^6 &= \beta^6\\
\beta^7 &= \beta^7\\
\beta^8 &= \beta^8\\
\beta^9 &= \beta^8 + \beta^7 + \beta^5 + \beta^4 + \beta + 1\\
\beta^{10} &= \beta^7 + \beta^6 + \beta^4 + \beta^2 + 1\\
\beta^{11} &= \beta^8 + \beta + \beta^5 + \beta^3 + \beta\\
\beta^{12} &= \beta^7 + \beta^6 + \beta^5 + \beta^2 + \beta + 1\\
\beta^{13} &= \beta^8 + \beta^7 + \beta^6 + \beta^3 + \beta^2 + \beta\\
\beta^{14} &= \beta^5 + \beta^3 + \beta^2 + \beta + 1\\
\beta^{15} &= \beta^6 + \beta^4 + \beta^3 + \beta^2 + \beta
\end{aligned}$$

$$\begin{aligned}
\beta^{16} &= \beta^7 + \beta^5 + \beta^4 + \beta^3 + \beta^2 \\
\beta^{17} &= \beta^8 + \beta^6 + \beta^5 + \beta^4 + \beta^3 \\
\beta^{18} &= \beta^8 + \beta^6 + \beta + 1 \\
\beta^{19} &= \beta^8 + \beta^5 + \beta^4 + \beta^2 + 1 \\
\beta^{20} &= \beta^8 + \beta^7 + \beta^6 + \beta^4 + \beta^3 + 1 \\
\beta^{21}(=\beta^0) &= 1.
\end{aligned}$$

Непримітивний код БЧХ, утворений з породжувального многочлена, який містить як базис степеневу послідовність β, β^2, β^3, β^4 базису β поля $GF(2^6)$ з періодом породжувального многочлена, що дорівнює 21, і степенем — 9, є кодом БЧХ (21, 12) DEC. Структуру цього коду наведено на рис. 52.

Інформаційні біти												Контрольні біти								
a_{11}	a_{10}	a_9	a_8	a_7	a_6	a_5	a_4	a_3	a_2	a_1	a_0	c_8	c_7	c_6	c_5	c_4	c_3	c_2	c_1	c_0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

$$\begin{aligned}
c_8 &= a_{11} + a_{10} + a_9 + a_8 + a_4 + a_2 + a_0 \pmod 2 \\
c_7 &= a_{11} + a_7 + a_4 + a_3 + a_2 + a_1 + a_0 \pmod 2 \\
c_6 &= a_{11} + a_9 + a_8 + a_6 + a_4 + a_3 + a_2 \pmod 2 \\
c_5 &= a_{10} + a_8 + a_7 + a_5 + a_3 + a_2 + a_0 \pmod 2 \\
c_4 &= a_{11} + a_{10} + a_8 + a_7 + a_6 + a_1 + a_0 \pmod 2 \\
c_3 &= a_{11} + a_8 + a_7 + a_6 + a_5 + a_4 + a_2 \pmod 2 \\
c_2 &= a_{10} + a_7 + a_6 + a_5 + a_4 + a_3 + a_1 \pmod 2 \\
c_1 &= a_9 + a_8 + a_5 + a_4 + a_3 + a_2 + a_0 \pmod 2 \\
c_0 &= a_{11} + a_{10} + a_9 + a_5 + a_3 + a_1 + a_0 \pmod 2
\end{aligned}$$

Рис. 52. Структура непримітивного коду БЧХ (21, 12) DEC

Коди БЧХ є типовими представниками кодів, що виправляють багатократні помилки циклічних кодів. Напрями створення циклічних кодів на базі породжувальних многочленів, які містять в собі коди БЧХ, є цілком зрозумілими.

БЧХ-коди — двійкові коди, в яких використовуються тільки значення "0" і "1", тобто поле $GF(2)$. Наприклад, у випадку використання як значення кодової мови поля $GF(2^8)$, це — багатозначний код БЧХ.

Різновидом такого багатозначного коду БЧХ є код Ріда—Соломона. Поява кодів Ріда—Соломона і їх щонайширше застосування в сучасній цифровій техніці стало, без перебільшення, видатною подією в теорії і практиці завадостійкого кодування.

Розділ 20

КОДИ РІДА–СОЛОМОНА

20.1. ВИЗНАЧЕННЯ КОДІВ РІДА–СОЛОМОНА

Після переходу на мову лінійної алгебри природно виникає бажання вивчати властивості лінійних кодів над іншими скінченними числовими полями. За допомогою такого поширення з'явилися коди Ріда—Соломона.

Код Ріда—Соломона був винайдений в 1960 р. співробітниками лабораторії Лінкольна Массачусетського технологічного інституту Ірвіном Рідом і Густавом Соломоном. Ідея використання цього коду наведена в статті "Polynomial Codes over Certain Finite Fields". Уперше код Ріда—Соломона застосовано у 1982 р. під час серійного випуску компакт-дисків. Взагалі, сфера їх використання дуже широка: кодери/декодери Ріда—Соломона можна знайти і в стрічкових запам'ятовувальних пристроях, і контролерах оперативної пам'яті, і модемах, і жорстких дисках, і приводах CD-ROM/DVD тощо. Ефективний алгоритм декодування запропоновано в 1969 р. Елвіном Берлекемпом і Джеймсом Мессі.

Коди Ріда—Соломона — це циклічні коди над числовим полем відмінним від $GF(2)$.

Нагадаємо, що існує безмежна кількість скінченних полів, і кількість елементів у такому полі завжди дорівнює степеню простого числа. Якщо ми зафіксуємо число елементів $n = qk$, то знайдеться єдине з точністю до ізоморфності скінченне поле з таким числом елементів, яке позначається як $GF(n)$. Проста реалізація цього поля — безліч многочленів за модулем многочлена $p(x)$ степеня k над незвідним полем залишків за модулем q. У разі многочленів з дійсними коефіцієнтами незвідними многочленами є тільки квадратні многочлени з негативним дискримінантом. Тому існує тільки квадратичне розширення дійсного поля — комплексні числа. А над скінченним полем існують незвідні многочлени будь-якого степеня. Зокрема, многочлен $g(z) = z^3 + z + 1$ — незвідний, і безліч многочленів за модулем $g(z)$ утворюють поле з $2^3 = 8$ елементів.

Отже, коди Ріда—Соломона або РС-коди (Reed-Solomon code, R-S code), належать до недвійкових циклічних кодів, тобто до кодів, символи яких узяті зі скінченного поля, що містить елементи $q > 2$ і яке позначається $GF(q)$ (q — степінь деякого простого числа). Елементами кодового вектора є не біти, а групи бітів (блоки). Вони дають змогу виправляти помилки в блоках даних. Дуже поширеними є коди Ріда—Соломона, що працюють з байтами (октетами).

Коди Ріда—Соломона — це дуже ефективні і зручні в реалізації (n, k) — блочні коди, що дають змогу виявляти і виправляти помилки в байтах. Вхідним словом у коді РС є блок з k байтів, вихідним — кодове слово з n байтів, що складається з k інформаційних і $n - k$ перевірних байтів. У цьому разі га-

рантовано, що під час декодування в кодовому слові виявляються і виправляються $t = \frac{n-k}{2}$ байтів незалежно від їх розташування усередині кодового слова. Параметр t називають корегувальною здатністю коду.

Нехай необхідно передати по каналу послідовність з M двійкових елементів вигляду

$$111 \ldots 1\ 101 \ldots 1\ 011 \ldots 0\ 100 \ldots 1.$$

Розіб'ємо цю послідовність на блоки по m елементів і позначимо їх деякими символами $\beta_0, \beta_1, \beta_2, \ldots, \beta_{n-1}$ ($n = \frac{M}{m}$). Повне число різних значень m-елементних блоків дорівнює $q = 2^m$.

Таким чином, послідовність, допередається, записується у вигляді деякої q-ї послідовності: $\beta_0, \beta_1, \beta_2, \ldots, \beta_{n-1}$. Деяка сукупність q-х послідовностей утворює q-й код. Такі коди, як і двійкові коди, можуть бути простими і завадостійкими.

Кодові комбінації q-го коду можна подати як многочлени з q-ми коефіцієнтами — елементами поля $GF(q)$. При цьому q-ті коефіцієнти як елементи поля $GF(q)$ є многочленами з двійковими коефіцієнтами. Наприклад,

$$B(x) = \beta_0(z)x^0 + b_1(z)x^1 + \ldots + \beta_{n-1}(z)x^{n-1},$$

де $\beta_i(z) = \beta_0 z^0 + \beta_1 z^1 + \ldots + \beta_{m-1} z^{m-1}$, $\beta_i = \{0,1\}$, z — формальна змінна многочлена з двійковими коефіцієнтами.

Інакше кажучи, коди Ріда—Соломона — це недвійкові циклічні коди, символами яких є m-бітові послідовності ($m > 2$ — додатне ціле число). Код (k, n) визначено на m-бітових символах для всіх n і k, для яких

$$0 < k < n < 2^m + 2, \tag{39}$$

де k — число інформаційних бітів, що підлягають кодуванню; n — число кодових символів у кодованому блоці. Для більшості згорткових кодів Ріда—Соломона (n, k)

$$(n, k) = (2^m - 1, 2^m - 1 - 2t). \tag{40}$$

Тут t — кількість помилкових бітів, які може виправити код; $n - k = 2t$ — число контрольних символів. Розширений код Ріда—Соломона можна отримати, якщо $n = 2^m$ або $2^m + 1$, але не більш того.

Код Ріда—Соломона має найбільшу мінімальну відстань, можливу для лінійного коду з однаковою довжиною вхідних і вихідних блоків кодера. Для недвійкових кодів відстань між двома кодовими словами визначається (за аналогією з відстанню Хеммінга) як число символів, якими розрізняються послідовності. Для кодів Ріда—Соломона мінімальна відстань визначається з рівняння

$$d_{\min} = n - k + 1. \tag{41}$$

Код РС виправляє всі спотворені символи, що містять помилку в t або меншому числі бітів (t — означено в рівнянні (43)). Число бітів t можна пода-

ти таким співвідношенням:

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor = \left\lfloor \frac{n-k}{2} \right\rfloor. \qquad (42)$$

Тут квадратні дужки означають найбільше ціле. Як бачимо з рівняння (42), коди Ріда—Соломона, які виправляють t символьних помилок, вимагають не більше ніж $2t$ контрольних символів, крім того, в декодері використовується $n - k$ надмірних символів, кількість яких удвічі перевищує кількість помилок, що виправляються. Код Ріда—Соломона, що виправляє t помилок, вимагає $2t$ перевірних символів і за його допомогою виправляються довільні пакети довжиною t символів і менше. Згідно з теоремою про межу Рейгера (а саме: кожен лінійний блоковий код, що виправляє всі пакети довжиною t і менше, повинен містити щонайменше $2t$ перевірних символів), коди Ріда—Соломона є оптимальними з погляду співвідношення довжини пакета і можливості виправлення помилок — використовуючи $2t$ додаткових перевірних символів, виправляють t помилок (і менше).

Для кожної помилки один надлишковий символ використовується для виявлення помилки, а другий — для визначення правильного значення.

Задачі, що виникають за наявності шумів і перешкод у каналі передачі, які викликають пакети помилок, діляться на дві основні групи. Тривалий час особливу увагу приділяли задачам першого типу, а саме: як подолати труднощі в отриманні м'яких рішень на виході демодулятора. У цьому випадку відсутня будь-яка інформація щодо окремих недостовірних символів. Якщо, проте, механізм виникнення пакетів помилок відомий, то отримані на підставі цього статистичні дані щодо помилок можна використовувати для їх виправлення. Задачі другого типу виникають у випадках, коли демодулятор може давати м'які рішення, що дозволяють виділити недостовірні символи. Якщо у разі передачі цих символів рівень шуму особливо великий, то вони іноді просто ***стираються***. У цьому випадку характеристики можна істотно поліпшити, оскільки є ефективні методи виправлення випадкових помилок і пакетів.

Задачі першого типу природно виникають у деяких цифрових системах зв'язку. Прикладами можуть бути системи зберігання інформації на магнітних і оптичних носіях. Пакети помилок у таких системах можуть зумовлюватися подряпинами, дефектами тощо. Унаслідок такого механізму виникнення помилок, отримати інформацію щодо достовірності кожного двійкового символу дуже важко або взагалі неможливо.

Типовий підхід полягає у розв'язанні цієї задачі методами, які дають змогу виправляти довгі пакети помилок за рахунок невиявлення деяких комбінацій випадкових помилок. У цьому разі можуть застосовуватися такі циклічні коди, як коди Файра і декодери Меггитта. Для виправлення декількох коротких пакетів можна використовувати коди Ріда—Соломона; у разі посимвольного чергування можна застосовувати такі самі коди для виправлення декількох довгих пакетів. Разом з відповідним чергуванням можна також використовувати блокові або згорткові коди, що виправляють випадкові помилки. Крім того, існують запроваджені Тонгом і Галлагером методи, які дають змогу виправляти довгі пакети помилок в припущенні, що між двома пакетами є досить довга ділянка, на якій немає помилок.

Інший підхід полягає у використанні дифузних згорткових кодів і порогового декодування, що дає змогу виправляти довгі пакети за наявності проміжних зон, які містять мало помилок. Головна проблема застосування всіх цих методів — отримання точної статистичної інформації про пакети в каналі. Часто одержати таку інформацію дуже складно. Наявність великої кількості методів кодування в подібних задачах спонукає не обговорювати їх детально.

Задачі другого типу є більш привабливими, а їх розв'язування за допомогою кодування — ефективнішим. Часто канал можна подати як канал з білим гауссівським шумом, в якому час від часу виникають значні шумові сплески або сплески унаслідок інтерференції, пов'язані, наприклад, із завмираннями або організованими перешкодами.

Припустимо, що використовується фазова модуляція. Близька до оптимальної стратегія отримання інформації про правдоподібність символів за наявності пакетів, спричинених інтерференцією, полягає у викреслюванні символів, пошкоджених інтерференцією, і оголошенні їх стертими.

Далі розглядатимемо саме таку стратегію, припускаючи, що довжина пакета більша, ніж тривалість одного символу, і що викреслюються тільки повні символи (цілком). Якщо, крім того, використовується чергування, то немає потреби у точній статистичній інформації про пакети (потрібно знати лише їх максимальну довжину). Зауважимо, що характеристики реальних систем виявляються дещо гіршими ніж теоретичні. Це зумовлено труднощами побудови ідеальних пристроїв стирання або використанням замість них простих обмежувачів.

Розглянемо декілька типів процесів, що призводять до виникнення шумових пакетів. За відсутності пакетів канал моделюється простим каналом з білим гауссівським шумом і характеризується деяким значенням відношення сигнал/шум. Передбачається, що за появи пакета шум є таким великим, що демодулятор його легко виявляє і викреслює відповідні символи. Як наслідок виникає пакет стертих символів. Вважатимемо, що або довжина пакета є фіксованою, або відома його максимальна довжина.

Щодо відстані між пакетами можна зробити декілька різних припущень. Одне з них полягає в тому, що пакети виникають ***періодично***. В цьому випадку пакет, що складається з B стертих символів, з'являється періодично через кожні P символів (параметр P називається ***періодом***, B — ***довжиною пакета***, $\delta = \frac{B}{P}$ — ***інтенсивністю пакета***). Типовий приклад виникнення такої ситуації — завмирання унаслідок інтерференції з імпульсним сигналом радіолокатора. У багатьох випадках такі пакети стирань призводять лише до незначного погіршення характеристик кодів, що виправляють випадкові помилки, навіть без чергування. Погіршення є істотним лише у разі дуже великої довжини пакета.

Суттєві труднощі виникають при ***випадкових*** пакетах стирань. У цьому випадку пакети складаються з фіксованого числа B символів із середньою інтенсивністю δ та з експоненціально розподіленими інтервалами часу між сусідніми пакетами. Оскільки тривалість інтервалів між пакетами є випадковою, лише дуже короткі пакети можуть виправлятися без перемежовування.

Найнеприємніший процес — ***організована перешкода***. За організованої перешкоди знання структури сигналів дає змогу здійснити стирання в найпроблемніших місцях. Наявність таких перешкод вимагає найбільш складних схем чергування.

Вірогідним також є виникнення ***випадкових стирань***. Цей процес вартий уваги, оскільки під час використання деяких схем чергування саме він визначає характеристики системи з кодуванням. Ці результати можуть бути підставою для порівняння з системою без чергування, що дає змогу з'ясувати обмінні співвідношення між додатковою перевагою унаслідок введення чергування і додатковим ускладненням системи.

Здатність коду до виправлення стирань виражається так:

$$p = d_{\min} - 1 = n - k. \tag{43}$$

Можливість одночасного виправлення помилок і стирань можна подати у вигляді наступної вимоги:

$$2\alpha + \gamma < d_{\min} < n - k. \tag{44}$$

Тут α — число символьних помилкових комбінацій, які можна виправити; γ — кількість комбінацій символьних стирань, які можуть бути виправлені.

Переваги недвійкових кодів, подібних до кодів Ріда—Соломона, можна з'ясувати з наступного порівняння. Розглянемо двійковий код $(n, k) = (7, 3)$. Повний простір n-бітових кодових слів містить $2^n = 2^7 = 128$ слів, із яких $2^k = 2^3 = 8$ (або 1/16 частина всіх n-бітових кодових слів) є інформаційними словами. Далі розглянемо недвійковий код $(n, k) = (7, 3)$, де кожен символ складається з $m = 3$ бітів. Простір nm-бітових кодових слів містить $2^{nm} = 2^{21} = 2\ 097\ 152$ слів, з яких $2^{km} = 2^9 = 512$ (або 1/4096 частина всіх nm-бітових слів) є кодовими словами. Якщо операції виконуються над недвійковими символами, кожний з яких утворений t бітами, то тільки незначна частина (тобто 2^{km} з великого числа 2^{nm}) можливих слів є інформаційними словами. Ця частина зменшується зі зростанням t. Тут важливим є наступне: якщо як інформаційні слова використовується незначна частина простору n-бітових кодових слів, то можна досягти більшого $d_{\min}$.

Будь-який лінійний код дає можливість виправити $n - k$ комбінацій символьних стирань, якщо всі $n - k$ стертих символів припадають на контрольні символи. Проте коди Ріда—Соломона мають чудову властивість — вони можуть виправити будь-який набір $n - k$ символів стирань у блоці. Можна сконструювати коди з будь-якою надлишковістю. Втім, зі збільшенням надлишковості зросте складність її високошвидкісної реалізації. Тому найбільш привабливі коди Ріда—Соломона мають дуже високий степінь кодування (низьку надлишковість).

Підсумовуючи сказане вище, кодами Ріда—Соломона можна називати циклічні (n, k)-коди за $n = q - 1$, безліч кодових комбінацій яких подається многочленами степеня $n - 1$ і менше з коефіцієнтами з поля $GF(q)$, ($q > 2$ і є степенем простого числа), а коренями породжувального многочлена є $n - k$ послідовних степенів: $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{d-1}$, деякого елемента α поля $GF(q)$ (d — мінімальна кодова відстань (n, k)-коду).

За означенням РС-код є підкласом кодів БЧХ з $m = 1$. Зазвичай елемент α вважають примітивним елементом поля, тобто всі степені α від першого до $(q - 1)$-го є різними ненульовими елементами поля $GF(q)$. Многочлен, що породжує РС-код, має степінь $n - k = d - 1$ і за теоремою Безу його можна визначити так:

$$g(x) = \prod_{t=1}^{d-1} \left(x - \alpha^t\right). \tag{45}$$

Відповідно до теорії циклічних кодів, породжувальний многочлен $g(x)$ є дільником x_{n-1} над полем $GF(q)$.

Таким чином, РС-код над полем $GF(q)$ має довжину кодової комбінації $n = q - 1$, число надмірних елементів у ній — $n - k = d - 1$, а мінімальна кодова відстань — $d = n - k + 1$.

Коди з подібним значенням мінімальної кодової відстані в теорії кодування називають ***максимальними***.

За фіксованих n і k не існує коду, мінімальна кодова відстань якого більша, ніж у РС-коду. Цей факт часто є вагомою підставою для використання РС-кодів. Водночас РС-коди завжди є коротшими за всі інші циклічні коди над тим самим алфавітом. РС-коди довжиною $n < q - 1$ називають ***укороченими***, а коди довжини q (або $q + 1$) — ***розширеними*** (подовженими) на один (або два) символи. У РС-коді може бути вибрано і інше значення m, якщо це виправдано.

Розглянемо деякі приклади на побудову РС-кодів.

Приклад 1. З'ясуємо, які РС-коди можна побудувати над розширеним полем $GF(2^2)$.

Визначаємо довжину кодової комбінації: $n = q - 1 = 3$. Задамося кодовою відстанню $d = 2$. Для реалізації такого РС-коду необхідна надлишковість $n - k = d - 1$. Тоді $k = 2$.

Якщо необхідно забезпечити $d = 3$, то слід задати надлишковість $n - k = 2$. Таким чином, над $GF(2^2)$ можна побудувати РС-коди (3, 2) з $d = 2$ і РС-коди (3, 1) з $d = 3$. Для РС-коду (3, 2) породжувальний многочлен на підставі визначення РС-коду подається так:

$$g(x) = x - \alpha.$$

Поле $GF(2^2)$ є розширенням поля $GF(2)$, тому знак "–" в $g(x)$ можна замінити на "+" як символ операції додавання в $GF(2)$, тобто слід прийняти, що $g(x) = x + \alpha$. Матриця, що породжує цей код, має вигляд

$$G(3,2) = \begin{bmatrix} \alpha & 1 & 0 \\ 0 & \alpha & 1 \end{bmatrix}.$$

РС-код (3, 2) над $GF(2^2)$ містить $q^k = 4^2 = 16$ дозволених комбінацій. Запишемо їх

0	0 0 0	4	0 α 1	8	0 α^2 α	12	0 1 α^2
1	α 1 0	5	α α^2 1	9	α α α	13	α 0 α^2
2	α^2 α 0	6	α^2 0 1	10	α^2 1 α	14	α^2 α^2 α^2
3	1 α^2 0	7	1 1 1	11	1 0 α	15	1 α α^2

Кожна така комбінація є многочленом другого або меншого степеня. Наведені вище послідовності коефіцієнтів кожної з кодових комбінацій задовольняють припущення, що коефіцієнти старших степенів розташовано справа, тобто інформаційні елементи займають дві позиції справа. При цьому надлишковий елемент займає позицію крайню зліва. Наприклад, комбінація 1 подається многочленом $\alpha + x$, комбінація 2 — многочленом $\alpha^2 + \alpha x$, ..., комбінація 14 — многочленом $\alpha^2 + \alpha^2 x + \alpha^2 x^2$.

Прослідкуємо формування надлишкових елементів для комбінацій 1 і 14:

Комбінація 1

$$\begin{array}{l} x \\ \oplus\ x+\alpha \\ \hline \end{array} \quad \begin{array}{|l} x+\alpha \\ \hline 1 \end{array}$$

α — *залишок*

Комбінація 14

$$\begin{array}{l} \alpha^2x^2+\alpha^2x \\ \oplus\ \alpha^2x^2+\alpha^3x \\ \hline \alpha^3x+\ \alpha^2x = x+\ \alpha^2x = \alpha x \\ \oplus \\ \hline \end{array} \quad \begin{array}{|l} x+\alpha \\ \hline \alpha^2x+\alpha \end{array}$$

$$\alpha x + \alpha^2$$

α^2 — *залишок*

Виконуючи дії над елементами поля $GF(2^2)$, корисно пам'ятати, що воно побудоване за модулем примітивного незвідного многочлена $\Pi(\alpha) = 1 + \alpha + \alpha^2 = 0$. Нижче наведено всі ненульові елементи цього поля, що є коренями многочлена $x^3 + 1$:

$$\alpha^0 = \alpha^3 = 10 = 1,$$
$$\alpha^1 = 01 = \alpha,$$
$$\alpha^2 = 11 = 1 + \alpha.$$

РС-код (3, 1) над полем $GF(2^2)$ з $d = 3$ містить чотири комбінації, кожна з яких включає в себе триразове повторення одного з елементів $GF(2^2)$:

$$000,\ 111,\ \alpha\alpha\alpha,\ \alpha^2,\ \alpha^2,\ \alpha^2.$$

(Показати справедливість цього твердження студенти повинні самостійно.)

Приклад 2. Побудуємо РС-код над простим полем $GF(5)$ з довжиною кодової комбінації $n = 4$ з відстанню $d = 3$.

Елементами поля $GF(5)$ — 0, 1, 2, 3, 4. Покажемо, що примітивним елементом цього поля є $\alpha = 2$:

$$2^0 = 1,$$
$$2^1 = 2,$$
$$2^2 = 4,$$
$$2^3 = 8 - 5 \cdot 1 = 3,$$
$$2^4 = 16 - 5 \cdot 3 = 1.$$

Як бачимо, чотири послідовні степені числа 2 дають усі ненульові елементи $GF(5)$, їх порядок дорівнює 4. Це доводить, що $\alpha = 2$ — примітивний елемент. Знайдемо породжувальний многочлен, степінь якого повинен бути $n - k = d - 1 = 2$. Для цього складаємо добуток

$$g(x) = (x - \alpha)(x - \alpha^0) = (x - 2)(x - 4) = x^2 - 6x + 8 = x^2 + (-6 + 5 \cdot 1) = x^2 + 4x + 3.$$

РС-код (4, 2) над $GF(5)$ має всього $qk = 5^2 = 25$ кодових комбінацій.

Нарешті, побудуємо матрицю цього породжувального коду, взявши за її рядки функцію $g(x)$ та її зсув,

$$G(4,2) = \begin{bmatrix} 3 & 4 & 1 & 0 \\ 0 & 3 & 4 & 1 \end{bmatrix}.$$

20.2. ВЛАСТИВОСТІ КОДІВ РІДА–СОЛОМОНА

20.2.1. Розширені коди Ріда–Соломона

Як відомо з теорії групових кодів, застосування до кодової комбінації додаткової перевірки на парність у багатьох випадках збільшує мінімальну відстань коду на одну одиницю. Для РС-кодів це явище спостерігається завжди. Нехай $C = c_0, c_1, \ldots, c_{n-1}$ кодова комбінація РС-коду з вагою d. Введемо додаткову перевірку на парність за правилом

$$C_n = -\sum_{i=0}^{n-1} c_i .$$

Покажемо, що мінімальна вага кодової комбінації C у разі одного розширення збільшується до $d + 1$. Це можливо за умови, якщо

$$C(1) = -C_n = \sum_{i=0}^{n-1} c_i \neq 0 . \qquad (46)$$

Проте $C(x) = \alpha(x)g(x)$ для деякого $\alpha(x)$ $a(x)$, тому $C(1) = \alpha(1)g(1)$. Очевидно, $g(1) \neq 0$. Крім того, $\alpha(1) \neq 0$, інакше $C(x)$ ділилося б на $(x - 1) \cdot g(x)$, тобто мало б вагу $d + 1$.

Приклад 3. Розширенням РС-коду (3, 2) $cg(x) = x + \alpha$ і $d = 2$ є РС-код (4, 2) з $d = 3$, породжувальний многочлен якого задається виразом $g(x) = (x + 1)(x + \alpha) = x^2 + \alpha^2 x + \alpha$, а породжувальна матриця має вигляд

$$G(4,2) = \begin{bmatrix} \alpha & \alpha^2 & 1 & 0 \\ 0 & \alpha & \alpha^2 & 1 \end{bmatrix}.$$

20.2.2. Укорочені коди Ріда–Соломона

РС-коди, як і будь-які групові коди, можна укорочувати за рахунок скорочення числа інформаційних елементів. Очевидно, що при цьому кодова відстань укороченого коду залишається точно такою самою, як і у початко-

вого коду $d = n - k + 1$. У загальному випадку укорочений РС-код, на відміну від вихідного, не є циклічним.

Існує спосіб побудови циклічного РС-коду над полем $GF(q)$ з довжиною кодової комбінації $n < q - 1$. З'ясуємо, як визначається породжувальний многочлен для такого РС-коду. Якщо α — примітивний елемент, то його порядок $l = q - 1$, і кожний ненульовий елемент $GF(q)$ можна знайти, як деякий степінь α. Порядок l_s кожного елемента $\alpha^s \le GF(q)$ є дільником $q - 1$, оскільки для кожного $\alpha^s \le GF(q)$ справедлива рівність

$$\left(\alpha^s\right)^{q-1} = 1. \tag{47}$$

Нехай в полі $GF(q)$ існує елемент α^s, порядок якого $1 < l_s < q - 1$. Тоді сукупність елементів $1, \alpha^s, \alpha^{2s}, \ldots, \alpha^{(l_s-1)^s}$ утворює підгрупу, яка складається зі всіх степенів одного з її елементів, тобто є циклічною і разом з нульовим елементом утворює підполе поля $GF(q)$, тобто є коренями многочлена $x^{l_s} - 1$.

Отже, справедливе таке співвідношення:

$$x^{l_s} - 1 = \prod_{i=1}^{l_s} \left(x - \alpha^{is}\right). \tag{48}$$

Таким чином, якщо в полі $GF(q)$ існує елемент α^s, порядок якого $1 < l_s < q - 1$, то можна побудувати циклічний РС-код над $GF(q)$ з довжиною кодової комбінації $n = l_s$ і породжувальним многочленом вигляду

$$g(x) = \prod_{i=1}^{l_s} \left(x - \alpha^{is}\right). \tag{49}$$

Приклад 4. У полі $GF(2^8)$ існує елемент α^{15}, порядок якого $l_{15} = 17$, отже, над $GF(2^8)$ можна побудувати РС-код з $n = 17$.

Інший спосіб отримання укорочених РС-кодів полягає в наступному. У виразі

$$x^{l_s} - 1 = \prod_{i=1}^{l_s} \left(x - \alpha^{is}\right) \tag{50}$$

здійснюємо підстановку $x \to x^m$, тоді

$$x^{ml_s} - 1 = \prod_{i=1}^{l_s} \left(x^m - \alpha^{is}\right). \tag{51}$$

Можна довести, що многочлен $x^m - \alpha^{is}$ має показник ml_s, з чого випливає, що за допомогою породжувального многочлена

$$g\left(x^m\right) = \prod_{i=1}^{d-1} \left(x^m - \alpha^{ls}\right) \tag{52}$$

може бути побудований РС-код з $n = ml_s$, який складається з m кодових комбінацій РС-кодів довжиною l_s, що чергуються.

20.2.3. Відображення кодів Ріда–Соломона над $GF(2^m)$ на двійкові коди

Елементи $GF(q)$ за $q = p^m$, як і вище, можуть бути подані послідовностями довжиною m з елементами поля $GF(p)$. У цьому випадку (n, k) — код Ріда—Соломона з відстанню d над $GF(p)$ є $(n_m = mn, k_m = mk)$-кодом над $GF(p)$ з відстанню $d_m \geq d$.

Якщо $q = 2^m$, то двійкові коди, отримані таким способом, часто мають велику мінімальну відстань.

Нехай $\xi_1, \ldots, \xi_m$ — базис елементів поля $GF(2^m)$ над $GF(2)$. Тоді, якщо $\beta = \sum_{i=1}^{m} b_i \xi_i$ — довільний елемент $GF(2^m)$, де $b_i = GF(2)$, то β відображається в послідовність довжиною m: $b_1, b_2, \ldots, b_m$.

Це відображення переводить лінійні коди в нелінійні. При цьому циклічні коди не обов'язково переходять у нециклічні.

Приклад 5. Використовуючи базис 1, α для елементів поля $GF(4)$ над $GF(2)$, отримуємо відображення $0 \to 00$, $1 \to 10$, $\alpha \to 01$, $\alpha^2 \to 11$. Тоді РС-код (3, 2) з $d = 2$ над полем $GF(4)$ (див. приклад 1) є двійковим (6, 4)-кодом з $d = 2$, наведеним у вигляді

0. 000000	4. 000110	8. 001101	12. 001011
1. 011000	5. 011110	9. 010101	13. 010011
2. 110100	6. 110010	10. 111001	14. 111111
3. 101100	7. 101010	11. 100001	15. 100111

Легко перевірити, що цей код не є циклічним.

Приклад 6. Розглянемо РС-код (7, 5) над $GF(2^3)$ з $d = 3$. Поле $GF(2^3)$, побудоване за модулем $\Pi(\alpha) = \alpha^3 + \alpha + 1$, містить вісім елементів

$$0\,0\,0 = 0, \quad 1\,0\,0 = 1, \quad 0\,1\,0 = \alpha, \quad 0\,0\,1 = \alpha^2,$$
$$1\,1\,0 = \alpha^3, \quad 0\,1\,1 = \alpha^4, \quad 1\,1\,1 = \alpha^5, \quad 1\,0\,1 = \alpha^6.$$

Як базис для елементів $GF(2^3)$ над $GF(2)$ використаємо $1, \alpha, \alpha^6$, тобто

$$\beta = b_1(100) + b_2(010) + b_3(101).$$

Тоді отримаємо відображення

$$0 \to 000,\ \alpha \to 010,\ \alpha^2 \to 101,\ \alpha^3 \to 110,\ \alpha^4 \to 111,\ \alpha^5 \to 001,\ \alpha^6 \to 001.$$

Сформуємо породжувальний многочлен у наступному вигляді:

$$g(x) = (x + \alpha^5)(x + \alpha^6) = \alpha^4 + \alpha x + x^2.$$

За такої побудови РС-код (7, 5) з $d = 3$ переходить у двійковий циклічний код (21, 15) з породжувальним многочленом $g(x) = 1 + x + x^2 + x^4 + x^6$ і мінімальною відстанню $d_{\min} = 3$. Це єдиний відомий нетривіальний приклад РС-коду, що відображається в двійковий циклічний код.

20.3. СПОСОБИ КОДУВАННЯ І ДЕКОДУВАННЯ КОДІВ РІДА–СОЛОМОНА

Способи кодування і декодування РС-кодів досить добре розроблені як теоретично, так і практично. Вони базуються на процедурах виправлення стирань помилок і сумісного виправлення помилок та стирань. Розглянемо процедури, що застосовуються в інформаційно-телекомунікаційній техніці.

20.3.1. Виправлення помилок

Почнемо розгляд з процедури декодування з виправленням помилок, відомий як ***алгоритм Форні***.

Нехай в приймач апаратури передачі даних (АПД) надійшла кодова комбінація РС-коду:

$$C(x) = f(x) + e(x),$$

де $f(x)$ — передана передавачем АПД кодова комбінація, в якій у процесі передачі по каналу зв'язку відбулося ν помилок, що відображаються многочленом $e(x)$ степеня ν. Кожен ненульовий компонент $e(x)$ описується парою елементів: Y_i — величиною помилки і X_i — номером позиції помилки (виявник або локатор помилки), Y_i, X_i — елементи $GF(q)$ і $X_i = \alpha^i$, $\alpha \in GF(q)$.

Для опису помилок використовуються:

1. Многочлен виявників (локаторів) помилок

$$\Lambda(x) = \prod_{i=1}^{\nu}(1 - xX_i) = \Lambda_\nu x^\nu + \Lambda_{\nu-1}x^{\nu-1} + \ldots + \Lambda_1 \cdot x + \Lambda_0, \qquad (53)$$

що має корені X_i^{-1}, $i = 1, \ldots, \nu$, відповідні до виявників помилок, тобто $X_i^{-1} \cdot \alpha_i = 1$.

2. Многочлен значень помилок

$$\Omega(x) = S(x)\Lambda(x)(\mathrm{mod}x^{2t}), \qquad (54)$$

де $S(x) = \sum_{j=1}^{2t} S_j x^j = \sum_{j=1}^{2t}\sum_{i=1}^{\nu} Y_i X_i^j x^j$ — синдромний многочлен нескінченного степеня, для якого відомі тільки $2t$ коефіцієнтів для комбінації РС-коду, що надійшла. Тут $S_j = \sum_{i=1}^{\nu} Y_i X_i^j = e\left(\alpha^j\right)$ — елемент синдрому; α^j — корінь породжувального многочлена РС-коду. Значення $S_j = e(\alpha^j)$ задаються перевірками:

$$S_j = C(\alpha^j) = f(\alpha^j) + e(\alpha^j) = e(\alpha^j), \qquad (55)$$

де $m_0 \le j \le m_0 + 2t - 1$ при $m_0 = 1$.

Рівність (57) визначається множиною з $(2t - \nu)$ рівнянь і називається ***ключовим рівнянням***, оскільки вона є "ключем" розв'язування задачі декодування. Це стає зрозумілим, якщо врахувати, що степінь $\Omega(x)$ не перевищує $\nu - 1$ і тому справедлива рівність

$$\left[\Lambda(x) \cdot S(x)\right]_\nu^{2t-1} = 0, \qquad (56)$$

де $\left[\alpha(x)\right]_m^n = \alpha_m x^m + \alpha_{m+1}x^{m+1} + \ldots + \alpha_n x^n$.

З ключового рівняння (54) необхідно отримати ν рівнянь для ν невідомих коефіцієнтів Λ_k. Ці рівняння є лінійними. Їх можна розв'язати звичайними методами або за допомогою ітераційного декодування, коли декодування здійснюється в декілька етапів, кожен з яких використовує інформацію від попереднього. Це дає можливість досягти значної ефективності, проте декодування вимагає великих ресурсів.

Визначивши многочлен $\Lambda(x)$, за ключовим рівнянням можна знайти невідомі компоненти вектора $e(x)$ і за ними вихідний вектор декодера $f(x) = C(x) + e(x)$.

Простим способом визначення коренів многочлена $\Lambda(x)$ є метод проб і помилок, відомий як ***процедура Ченя***. Ця процедура полягає в перебиранні всіх можливих варіантів, а також у послідовному обчисленні $\Lambda(\alpha^j)$ для кожного j і перевірці одержаних значень на рівність нулю. Усі 2^m можливих символів один за одним підставляються почергово в поліном виявника помилок. Далі виконується розрахунок многочлена. Якщо результат перетворюється на нуль — вважається, що корені знайдено. Таким чином, якщо величина $\Lambda(\alpha^{-k})$ дорівнює нулю, то α^k відповідає кореню многочлена виявників помилок і k-й елемент кодової комбінації містить помилку. Найпростішим способом обчислення значення $\Lambda(x)$ в точці β є ***процедура Горнера***:

$$\Lambda(\beta) = (...(((\Lambda\beta + \Lambda_{\nu-1})\beta_\sigma + \Lambda_{\nu-2})\beta_\sigma + \Lambda_{\nu-3})\beta + ... + \Lambda_0). \qquad (57)$$

Для обчислення $\Lambda(\beta)$ за процедурою Горнера (57) потрібно тільки σ множень і ν додавань (ν — степінь $\Lambda(x)$).

Визначивши виявники помилок за допомогою ключового рівняння (54), знайдемо значення помилок. Для цього, використовуючи значення співмножників, що входять у рівність для $\Omega(x)$, запишемо ключове рівняння так:

$$\Omega(x) = \left[\sum_{j=1}^{2t}\sum_{i=1}^{\nu} Y_i X_i^j x^{j-1}\right]\left[\prod_{i=1}^{\nu}(1 - X_i x)\right]\left(\mathrm{mod}\, x^{2t}\right) =$$
$$= \sum_{i=1}^{\nu} Y_i X_i \left[(1 - X_i x)\sum_{j=1}^{2t}(X_i x)^{j-1}\right]\prod_{j\neq i}(1 - X_i x)\left(\mathrm{mod}\, x^{2t}\right). \qquad (58)$$

Згортаючи вираз в квадратних дужках, остаточно отримуємо

$$\Omega(x) = \sum_{i=1}^{\nu} Y_i X_i \left(1 - X_i^{2t} x^{2t}\right)\prod_{j\neq i}(1 - X_i x)\left(\mathrm{mod}\, x^{2t}\right). \qquad (59)$$

Зводячи цей вираз за модулем x^{2t}, записуємо

$$\Omega(x) = \sum_{i=1}^{\nu} Y_i X_i \prod_{j\neq i}\left(1 - X_i^{2t} x^{2t}\right). \qquad (60)$$

Обчислимо многочлен значень помилок на позиції $l : x = X_i^{-1}$:

$$\Omega\left(X_i^{-1}\right) = \sum_{i=1}^{\nu} Y_i X_i \prod_{j\neq i}\left(1 - X_j X_i^{-1}\right).$$

Звідси

$$Y_l = \frac{\Omega\left(X_l^{-1}\right)X_l^{-1}}{\prod_{j\neq l}\left(1-X_jX_l^{-1}\right)}.$$

Знайдемо похідну від многочлена виявників помилок $L(x)$:

$$\Lambda'(x) = -\sum_{i=1}^{v} X_i \prod_{j\neq i}\left(1-xX_j\right).$$

Для l-ї позиції отримуємо

$$\Lambda'\left(X^{-1}\right) = -X_l \prod_{j=l}\left(1-X_jX_l^{-1}\right).$$

З урахуванням останнього виразу Y_l значення помилки на позиції l набуває вигляду

$$Y_l = -\frac{\Omega\left(X_l^{-1}\right)}{\Lambda'\left(X_l^{-1}\right)}.$$

Такий спосіб обчислення значення помилки відомий в літературних джерелах як алгоритм Форні. Він дає змогу визначити значення помилок за відомими многочленами виявників і значень помилок.

Наведені многочлени обчислюються за результатами розв'язування ключового рівняння.

20.4. АЛГОРИТМ БЕРЛЕКЕМПА–МЕССІ РОЗВ'ЯЗУВАННЯ КЛЮЧОВОГО РІВНЯННЯ

У разі декодування як у часовій, так і в частотній області доводиться знаходити многочлен виявників помилок $\Lambda(x)$ як многочлен найменшого степеня, для якого виконується ключове рівняння (54). Крім того, у разі декодування в часовій області потрібно також визначити многочлен значень помилок $\Omega(x)$. Якщо інтерпретувати $\Lambda(x)$ як многочлен, який задає зворотні зв'язки в лінійному регістрі зсуву, то розв'язування ключового рівняння еквівалентне знаходженню регістра зсуву зі зворотними зв'язками найменшої довжини, який формує перші члени многочлена синдрому. На практиці обидва ці підходи є дієвими.

У перших наведених у літературних джерелах процедурах розв'язування ключового рівняння використовувалися стандартні методи обернення матриць. Проте пізніше Берлекемп запропонував просту ітеративну схему, яка на сьогодні використовується в більшості практичних застосувань. Також показано, що розв'язок можна знайти за допомогою алгоритму Евкліда. Одна з переваг цього алгоритму — те, що для сприйняття він простіший, ніж алгоритм Берлекемпа. Водночас він тісно пов'язаний з цим алгоритмом. Тому стисло викладемо обидва алгоритми.

Алгоритм Евкліда — рекурентний метод визначення найбільшого загального дільника двох цілих чисел або двох многочленів. Основне твердження,

на якому базується цей алгоритм, полягає в наступному. Нехай a і b — два цілі числа або два многочлени, причому $a \gg b$, якщо це числа, і $\deg(a) \geq \geq \deg(b)$, якщо це многочлени (deg означає степінь многочлена). Поділимо a на b. Якщо залишок r від ділення дорівнює "0", то $d = b$ — найбільший загальний дільник. Якщо залишок не дорівнює "0", то замінимо a на b, b на r і повторимо ділення. Наступний простий приклад показує, що за цим алгоритмом дійсно розраховується найбільший спільний дільник.

Приклад 7. Припустимо, що дано два цілі числа $a = 186$ і $b = 66$, тоді

$$186 = 66 \cdot 2 + 54,$$
$$66 = 54 \cdot 1 + 12,$$
$$54 = 12 \cdot 4 + 6,$$
$$12 = 6 \cdot 2 + 0.$$

Оскільки d ділить 186 і 66, воно має ділити і 54; оскільки d ділить 66 і 54, воно повинне ділити 12, нарешті, оскільки воно ділить 54 і 12, воно має ділити 6, і тому є найбільшим спільним дільником. У процесі знаходження найбільшого спільного дільника, згідно з алгоритмом, обчислюються два числа або многочлени f і g, для яких $fa + gb = d$.

Проте кориснішим є не кінцеві результати, а проміжні. За кожної ітерації обчислюються числа або многочлени f_i, g_i такі, що

$$f_i a + g_i b = r_i. \tag{61}$$

Рекурентні співвідношення між f_i, g_i, r_i стають зрозумілими, якщо ще раз звернутися до наведеного прикладу.

Перше рівняння має, очевидно, потрібний вигляд. Отже, за першої ітерації алгоритму отримуємо $1 \cdot 186 - 2 \cdot 66 = 54$.

Друге рівняння (див. приклад 7) містить вказівки про те, що відбувається за другої ітерації: віднімаючи попереднє рівняння з рівняння $66 = 66$, отримуємо $(-1) \cdot 186 + 3 \cdot 66 = 12$.

Аналогічно після третьої і четвертої ітерацій знаходимо відповідно

$$5 \cdot 186 = 14 \cdot 66 + 6,$$
$$(-11) \cdot 186 + 31 \cdot 66 = 0.$$

Зауважимо, що кожне наступне рівняння одержуємо з двох попередніх за формулою

$$Eq(i) = Eq(i-2) - q_i Eq(i-1),$$

де q_i — відношення двох попередніх залишків.

На відміну від прикладу 7, надалі нас цікавитимуть головним чином многочлени, тому всі подальші результати стосуватимуться цього випадку.

Визначимо $q_i(x)$ як частку від ділення $r_{i-2}(x)$ на $r_{i-1}(x)$, тобто покладемо, що

$$q_i(x) = \left[\frac{r_{i-2}(x)}{r_{i-1}(x)} \right]_0^{\infty}, \tag{62}$$

де квадратна дужка означає додатні степені x. Тоді

$$\begin{aligned} r_i(x) &= r_{i-2}(x) - q_i(x)r_{i-1}(x), \\ f_i(x) &= f_{i-2}(x) - q_i(x)f_{i-1}(x), \\ g_i(x) &= g_{i-2}(x) - q_i(x)g_{i-1}(x). \end{aligned} \tag{63}$$

Початковими значеннями є

$$\begin{aligned} f_{-1}(x) &= g_0(x) = 1, \\ f_0(x) &= g_{-1}(x) = 0, \\ r_{-1}(x) &= a(x), \\ r_0(x) &= b(x). \end{aligned} \tag{64}$$

Алгоритм побудовано так, що

$$\deg\left[r_i(x)\right] < \deg\left[r_{i-1}(x)\right], \tag{65}$$

для всіх i.

Отже, після скінченного числа ітерацій залишок обов'язково буде дорівнювати 0. Простий приклад з $a(x) = x^3 + 1$ і $b(x) = x^2 + 1$ наведено в табл. 32. Останній ненульовий залишок $r_i(x) = x + 1$ є найбільшим спільним дільником.

Наша задача, проте, полягає не в знаходженні найбільшого спільного дільника, а у розв'язуванні ключового рівняння.

Зв'язок алгоритму Евкліда з цією задачею стане зрозумілим, коли рівняння (64) запишемо у вигляді

$$g_i(x)b(x) \equiv r_i(x) \bmod a(x). \tag{66}$$

Вважаючи, що $a(x) = x^{2t}$, отримуємо

$$g_i(x)b(x) \equiv r_i(x) \bmod x^{2t}, \tag{67}$$

тобто ключове рівняння з $g_i(x) = \Lambda_i(x)$ і $r_i(x) = \Omega_i(x)$.

Таким чином, алгоритм Евкліда дає послідовність розв'язків ключового рівняння.

Щоб показати, як алгоритм Евкліда приводить до необхідного розв'язування ключового рівняння, потрібно використати його властивість, яка виражається такою рівністю:

$$\deg[g_i(x)] + \deg[r_{i-1}(x)] = \deg[a(x)]. \tag{68}$$

Вважаючи, що $a(x) = x^{2t}$, отримуємо

$$\begin{aligned} \deg[g_i(x)] + \deg[r_{i-1}(x)] &= 2t, \\ \deg[g_i(x)] + \deg[r_i(x)] &< 2t. \end{aligned} \tag{69}$$

Таблиця 32. Приклад алгоритму Евкліда для $a(x) = x^3 + 1$ і $b(x) = x^2 + 1$

i	$f_i(x)$	$g_i(x)$	$r_i(x)$	$q_i(x)$
–1	1	0	$x + 1$	—
0	0	1	$x + 1$	—
1	1	x	$x + 1$	x
2	$x + 1$	$x^2 + x + 1$	0	$x + 1$

Нагадаємо, що при появі числа помилок меншого ніж або рівного t для розв'язку, що нас цікавить, маємо $\deg[\Omega(x)] < \deg[\Lambda(x)] \le t$. Існує єдиний (із точністю до сталого множника) многочлен $\Lambda(x)$, степінь якого не перевищує t, що задовольняє ключове рівняння. Крім того, якщо $\deg[r_{i-1}(x)] \ge t$, отже, $\deg[g_i(x)] \le t$ і $\deg[r_i(x)] < t$, то $\deg[g_{i+1}(x)] > t$. Тому проміжні результати на i-му кроці дають єдиний розв'язок ключового рівняння, що нас цікавить. Таким чином, щоб розв'язати ключове рівняння слід застосовувати алгоритм Евкліда доти, поки не буде виконано умову $\deg[r_i(x)] < t$.

Наведений опис методу розв'язування ключового рівняння узагальнюємо спочатку для кодів БЧХ, що виправляють t помилок:

- застосовуємо алгоритм Евкліда до $a(x) = x^{2t}$ і $b(x) = S(x)$;
- використовуємо початкові умови (67);
- зупиняємося, якщо $\deg[r_n(x)] < t$;
- покладаємо, що $\Lambda(x) = g_n(x)$ і $\Omega(x) = r_n(x)$.

Застосування такого алгоритму проілюструємо на двох прикладах. Усі арифметичні операції виконуються в полі $GF(2^4)$ з використанням табл. 33.

Приклад 8. Розглянемо двійковий код БЧХ довжиною 15, що виправляє потрійні помилки. Покладемо, що $m_0 = 1$, тому коренями породжувального многочлена є елементи α, α^2, ..., α^6, де α — примітивний елемент $GF(2^4)$. Припустимо, що прийняте слово має вигляд

$$r(x) = x^2 + x^{10}, \tag{70}$$

і що здійснюється декодування в часовій області. Тоді, обчислюючи шукані коефіцієнти синдрому за формулою

$$S_j = r_{n-1}\alpha^{(n-1)(j+m_0)} + r_{n-2}\alpha^{(n-2)(j+m_0)} + \ldots + r_0,$$

отримуємо

$$S_j = \left(\alpha^{j+1}\right)^3 + \left(\alpha^{j+1}\right)^{10}, \quad j = 0, 1, \ldots, 5.$$

Звідси многочлен синдрому —

$$S(x) = \alpha^{14}x^5 + \alpha^{10}x^4 + \alpha^3x^3 + \alpha^7x^2 + \alpha^9x + \alpha^{12}. \tag{71}$$

Починати пошук розв'язку можна з $a(x) = x^6$ і $b(x) = S(x)$.

Таблиця 33. Звичайне і логарифмічне подання ненульових елементів поля $GF(2^4)$ для $p(x) = x^4 + x + 1$

i	α^i	i	α^i
0	0 0 0 1	8	0 1 0 1
1	0 0 1 0	9	1 0 1 0
2	0 1 0 0	10	0 1 1 1
3	1 0 0 0	11	1 1 1 0
4	0 0 1 1	12	1 1 1 1
5	0 1 1 0	13	1 1 0 1
6	1 1 0 0	14	1 0 0 1
7	1 0 1 1		

Таблиця 34. Розв'язки ключового рівняння для $S(x)$, що задається рівнянням (71)

i	$\Lambda_i(x)$	$\Omega_i(x)$	$q_i(x)$
–1	0	x^6	—
0	1	$S(x)$	—
1	$\alpha x + \alpha^{12}$	$\alpha^3x^4 + \alpha^2x^3 + \alpha^2x^2 + x + \alpha^9$	$\alpha x = \alpha^{12}$
2	$\alpha^{12}x^2 + \alpha^{11}x + \alpha^{14}$	α^{14}	$\alpha^{11}x + \alpha^6$

Алгоритм Евкліда, за допомогою якого знаходять розв'язки ключового рівняння, і многочлен виявників помилок, наведено в табл. 34.

Зауважимо, що розв'язок відповідає значенню i, для якого $\deg[\Omega_i(x)] < t = 3$. Коренями отриманого многочлена виявників помилок $\Lambda(x) = \alpha^{12}x^2 + \alpha^{11}x + \alpha^{14}$ є α^{-3} і α^{-10}; їх можна знайти або безпосередньою підстановкою всіх елементів $GF(2^4)$, або за допомогою процедури Ченя. Таким чином, гіпотетична комбінація двох помилок має вигляд $\hat{e}(x) = x^3 + x^{10}$, а слово на виході декодера — $\hat{c}(x) = r(x) - \hat{e}(x) = 0$.

При декодуванні в частотній області многочлен виявників помилок використовується по-іншому. Многочлен $\Lambda(x)$ указує відведення регістра зсуву зі зворотними зв'язками, за допомогою яких можна отримати перетворення вектора помилок, що задовольняє рівняння

$$\sum_{k=0}^{n-1} \Lambda_k E_{j-k} = 0, \tag{72}$$

для заданого ν-компонентного блока. Для наведеного прикладу вираз (72) можна записати так:

$$\hat{E}_j = \alpha^{12}\hat{E}_{j-1} + \alpha^{13}\hat{E}_{j-2}.$$

Відомими є значення $\hat{E}_j = S_{j-1}$ (j = 1, 2, ...,6). Рекурентна процедура дає перетворення вектора помилок у вигляді

$$\hat{E} = (0, \alpha^{12}, \alpha^9, \alpha^7, \alpha^3, \alpha^{10}, \alpha^{14}, \alpha^7, \alpha^6, \alpha^{11}, \alpha^5, \alpha^{11}, \alpha^{13}, \alpha^{13}, \alpha^{14}).$$

Виконавши зворотне перетворення, отримаємо гіпотетичний вектор помилок:

$$\hat{e} = (0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0).$$

Здійснюючи вказані виправлення, знову одержуємо нульове кодове слово.

Приклад 9. Розглянемо код Ріда—Соломона довжиною 15 над полем $GF(2^4)$, що виправляє потрійні помилки. Коренями породжувального многочлена є α, α^2, ..., α^6. Припустимо, що прийняте слово має вигляд

$$r(x) = \alpha^7x^3 + \alpha^{11}x^{10}$$

і виконується декодування в часовій області. Коефіцієнти синдрому можна обчислити за формулою

$$S_j = \alpha^7\left(\alpha^{j+1}\right)^3 + \alpha^{11}\left(\alpha^{j+1}\right)^{10}, j = 0, 1, ..., 5.$$

Отриманий многочлен синдрому має вигляд

$$S(x) = \alpha^{14}x^5 + \alpha^{14}x^4 + \alpha^{12}x^3 + \alpha^6x^2 + \alpha^{12}x + \alpha^7. \quad (73)$$

Алгоритм Евкліда, за допомогою якого знаходять многочлени виявників і значення помилок, наведено в табл. 35.

Коренями отриманого многочлена виявників помилок

$$\Lambda(x) = \alpha^9x^2 + \alpha^8x + \alpha^{11}$$

є α^{-2} і α^{-10}. Щоб знайти значення помилок в α^{-2} і α^{-10} за формулою

$$e_{i_m} = \frac{\Omega\left(\alpha^{-i_m}\right)}{\Lambda\left(\alpha^{-i_m}\right)}, \quad m_0 = 1, \quad (74)$$

можна використовувати многочлен значень помилок

$$\Omega(x) = \alpha^2x + \alpha^2$$

і похідну $\Lambda'(x)$:

$$\Lambda'(x) = \alpha^2.$$

Одержуємо $\hat{e}(x) = \alpha^7x^3 + \alpha^{11}x^{10}$, і відповідним словом на виході декодера буде

$$\hat{c}(x) = r(x) - \hat{e}(x) = 0.$$

Підводячи підсумок, зазначимо, що унаслідок застосування алгоритму Евкліда отримаємо послідовність розв'язування ключового рівняння, причому степені $\Lambda_i(x)$ зростають, а степені $\Omega_i(x)$ спадають. Проте, як було показано, для будь-якої комбінації з t або менше помилок у коді $\deg[\Omega(x)] < \deg[\Lambda(x)]$. Тому єдиний розв'язок, що гарантує виправлення помилок потрібної кратності, відповідає першому i, для якого $\deg[\Omega_i(x)] < t$.

Алгоритм Берлекемпа найлегше, мабуть, зрозуміти при наступному підході. Знаходження розв'язку ключового рівняння, що має мінімальний степінь, еквівалентне побудові регістра зсуву мінімальної довжини із зворотними зв'язками, які відображають $\Lambda(x)$, що породжують перші $2t$ членів $S(x)$. Основний зміст алгоритму Берлекемпа—Мессі формулюється таким чином. Спочатку визначають найкоротший регістр зсуву, що генерує символ S_1. Далі перевіряють, чи породжує цей регістр також символ S_2. Якщо породжує, то даний регістр, як і раніше, залишається якнайкращим розв'язком. Потім потрібно перевірити, чи породжує він наступні символи синдромного многочлена. На деякому кроці черговий символ вже не генеруватиметься. У цей момент потрібно змінити регістр так, щоб він:

- правильно передбачав наступний символ;

Таблиця 35. Розв'язки ключового рівняння для $S(x)$, що задається рівнянням (73)

i	$\Lambda_i(x)$	$\Omega_i(x)$	$q_i(x)$
–1	0	x^6	—
0	1	$S(x)$	—
1	$\alpha x + \alpha$	$\alpha^6x^4 + \alpha^5x^3 + \alpha^5x^2 + \alpha^3x + \alpha^8$	$\alpha x + \alpha$
2	$\alpha^{12}x^2 + \alpha^{11}x + \alpha^{14}$	α^{11}	$\alpha^{11}x + \alpha^6$

• не змінював прогноз попередніх символів;
• збільшував довжину регістра на мінімально можливе значення.

Процес обчислення продовжується доти, поки не будуть породжені перші $2t$ символів синдрому.

Алгоритм будується за ітераційною процедурою. За кожної ітерації повинні зберігатися як многочлен зв'язків $\Lambda(x)$, так і додатки $B(x)$. Для кожного нового члена $S(x)$ передбачається перевірка правильності прогнозу цього символу поточним многочленом зв'язку. Якщо прогноз правильний, то многочлен зв'язків не змінюється, а додаток множиться на x. Якщо прогноз неправильний, то поточний многочлен зв'язків змінюють, додаючи до нього додаток. Потім перевіряють, чи збільшилася довжина регістра. Якщо вона не збільшилася, то поточний додаток залишають, а якщо — збільшилася, то кращим додатком вважають попередній многочлен зв'язків. Для недвійкових кодів додаток нормують, щоб незв'язність дорівнювала 1. Далі у разі кожного виправлення такий нормалізований додаток множать на значення поточної незв'язності.

Блок-схему розв'язування ключового рівняння за алгоритмом Берлекемпа—Мессі наведено на рис. 53, а процедуру декодування кодів Ріда—Соломона з виправленням помилок, що отримала назву швидкого декодування, — на рис. 54. Для виправлення помилок декодер виконує послідовність обчислень, що реалізують алгоритм Берлекемпа—Мессі і алгоритм Форні.

Алгоритм виконується за $2t = n - k$ ітерацій. При цьому многочлен значень помилок знаходиться безпосереднім множенням за формулою ключового рівняння (54).

На рис. 53 введено такі позначення: r — номер ітерації, $r = 0, 1, 2, ..., n - k = 2t$; S_r — компонент синдрому; Δr — помилка в обчисленні S_r (r-та незв'язність); $\Lambda(x)$ — многочлен виявників помилок, відповідно до компонентів якого будується регістр зсуву мінімальної довжини; L — довжина регістра зсуву, $L \geq \deg \Lambda(x)$; $B(x)$ — нормувальний додаток; $\Omega(x)$ — многочлен значень помилок.

Існує доповнення до алгоритму Берлекемпа—Мессі, що дає змогу в ході розв'язування ключового рівняння одночасно із визначенням значення многочлена виявників помилок $\Lambda(x)$ степеня не вищого ніж t знаходити многочлен значень помилок $\Omega(x)$ степеня не вищого ніж $t - 1$. Це доповнення на рис. 53 зображене окремими блоками.

Оскільки $\Omega(x)$ визначається множенням $\Lambda(x)$ на $S(x)$, можна задати послідовність многочленів $\Omega_r(x)$, що задовольняють ті самі рекурентні співвідношення, що і многочлени $\Lambda_r(x)$. Щоб уникнути неоднозначності, позначимо додаток при знаходженні $\Omega(x)$ через $A_r(x)$. Тоді

$$\Omega_{r+1}(x) = \Omega_r(x) + \Delta r A_r(x)$$

$$A_{r+1}(x) = \begin{cases} xA_r(x), & \text{якщо } L_{r+1} = L_r, \\ x\Omega_r(x)\Delta r^{-1}, & \text{якщо } L_{r+1} > L_r, \end{cases}$$

При цьому вважається, що $\Omega_0(x) = 0$ і $A_0(x) = 1$.

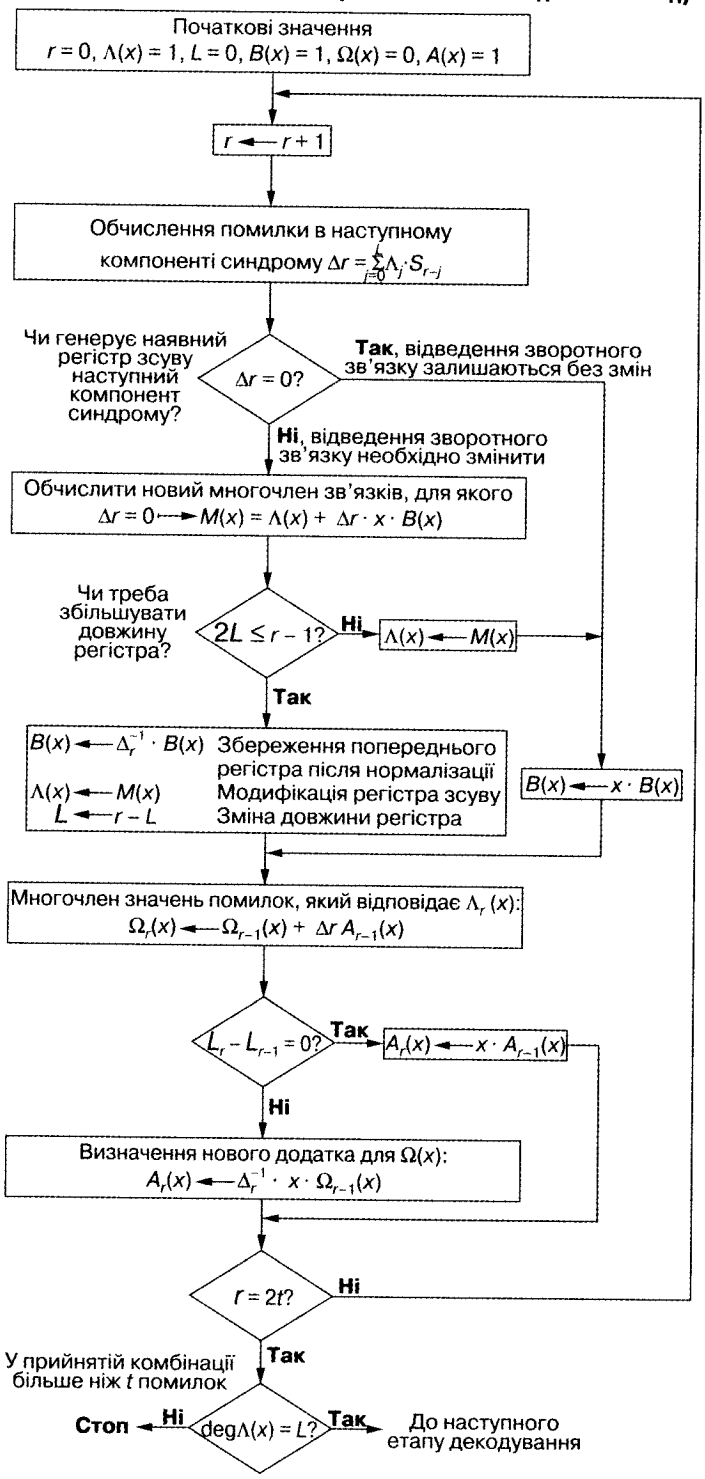


Рис. 53. Блок-схема розв'язування ключового рівняння за алгоритмом Берлекемпа—Мессі

Приклад 10. Розглянемо код Ріда—Соломона (15, 9) над полем $GF(2^4)$. Число надмірних елементів в кодовій комбінації $n - k = 6$. Мінімальна кодова відстань $d = n - k + 1 = 7$. Кратність гарантованих помилок t, що виправляються, — $t = 3$.

Корені породжувального многочлена: α, α^2, α^3, α^4, α^5, α^6 — елементи $GF(2^4)$. Запишемо породжувальний многочлен

$$g(x) = (x + \alpha)(x + \alpha^2)(x + \alpha^3)(x + \alpha^4)(x + \alpha^5)(x + \alpha^6) =$$
$$= x^6 + \alpha^{10}x^5 + \alpha^{14}x^4 + \alpha^4x^3 + \alpha^6x^2 + \alpha^9x + \alpha^6.$$

Тут і далі під час обчислень використано таблицю додавання в полі $GF(2^4)$. Нехай від передавача в канал передано кодовану послідовність $f(x) = 0$. Приймач приймає послідовність

$$C(x) = \alpha x^7 + \alpha^5x^5 + \alpha^{11}x^2 = e(x).$$

Компоненти синдрому обчислюються за формулою

$$S_j = \alpha(\alpha^j)^7 + \alpha^5(\alpha^j)^5 + \alpha^{11}(\alpha^j)^2,$$

де $j = 1, 2, ..., 6$.

Синдромний многочлен має вигляд

$$S(x) = \alpha^{11}x^6 + \alpha^0x^5 + \alpha^{13}x^4 + \alpha^{14}x^3 + \alpha^0x^2 + \alpha^{12}x^1.$$

Розв'язавши ключове рівняння за алгоритмом Берлекемпа—Мессі, отримаємо:

- многочлен виявників помилок

$$\Lambda(x) = 1 + \alpha^{14}x + \alpha^{11}x^2 + \alpha^{14}x^3 =$$
$$= (1 + \alpha^7x)(1 + \alpha^5x)(1 + \alpha^2x);$$

- многочлен значень помилок

$$\Omega(x) = \alpha^{12} + \alpha^{12}x + \alpha^8x^2.$$

Послідовність виконання алгоритму наведено на рис. 55.

Важливо те, що отриманий многочлен $\Lambda(x)$ містить як виявники помилок X_2, X_5, X_7, що відповідає $e(x)$.

Перевіримо правильність обчислення $\Omega(x)$. Для цього одержимо многочлен значень помилок безпосереднім обчисленням:

$$\Omega(x) = S(x) \cdot \Lambda(x).$$

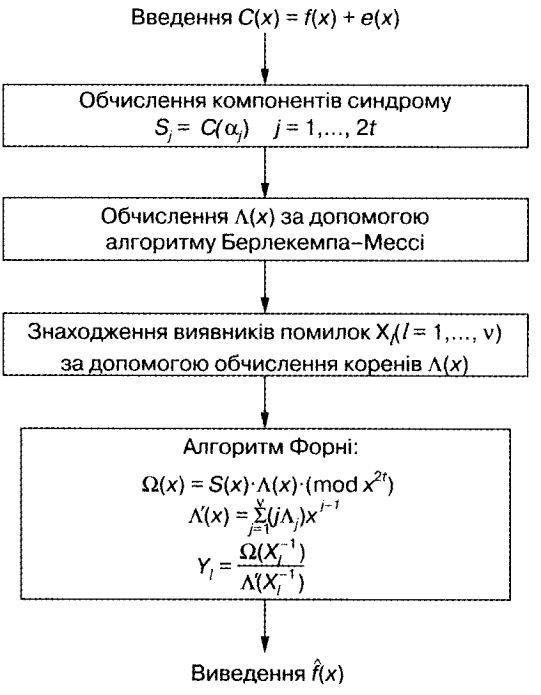


Рис. 54. Блок-схема швидкого декодування кодів Ріда—Соломона

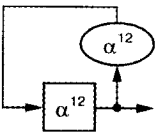
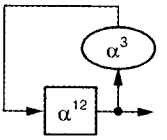
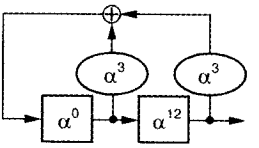
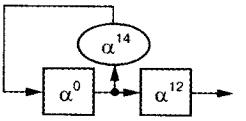
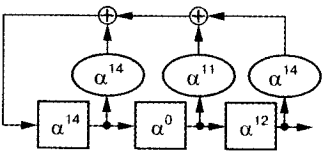
Крок ітерації	Многочлен зв'язків $L_r(x)$	Довжина регістра L_r	Регістр зсуву із зворотними зв'язками (з початковим завантаженням)	Значення $S(x)$, що генерується
1	$1 + \alpha^{12}x$	1		$S_1 = \alpha^{12}$
2	$1 + \alpha^{3}x$	1		$S_1 = \alpha^{12}$, $S_2 = \alpha^{0}$
3	$1 + \alpha^{3}x + \alpha^{3}x^2$	2		$S_1 = \alpha^{12}$, $S_2 = \alpha^{0}$, $S_3 = \alpha^{14}$
4	$1 + \alpha^{14}x$	2		$S_1 = \alpha^{12}$, $S_2 = \alpha^{0}$, $S_3 = \alpha^{14}$, $S_4 = \alpha^{13}$
5	$1 + \alpha^{14}x + \alpha^{11}x^2 + \alpha^{14}x^3$	3		$S_1 = \alpha^{12}$, $S_2 = \alpha^{0}$, $S_3 = \alpha^{14}$, $S_4 = \alpha^{13}$, $S_5 = \alpha^{0}$, $S_6 = \alpha^{11}$

Рис. 55. Побудова регістра мінімальної довжини, що породжує $S(x)$, за алгоритмом Берлекемпа—Мессі (приклад 10)

Для виконання множення многочленів подамо $S(x)$ у вигляді, що відповідає $\Lambda(x)$:

$$S(x) = \alpha^{12} + x + \alpha^{14}x^2 + \alpha^{13}x^3 + x^4 + \alpha^{11}x^5.$$

Обмежуючи кількість членів добутку п'ятим степенем (за mod $x^{2t=6}$), отримуємо

$$\Omega(x) = \alpha^{12} + \left(\alpha^{11} + 1\right)x + (\alpha^{8} + \overbrace{\alpha^{14} + \alpha^{14}}^{=0})x^2 + \overbrace{(\alpha^{11} + \alpha^{11} + \alpha^{13} + \alpha^{13})}^{=0}x^3 +$$

$$+ \underbrace{\left(\alpha^{14} + \alpha^{10} + \alpha^{12} + 1\right)}_{=0}x^4 + \underbrace{\left(\alpha^{13} + \alpha^{9} + \alpha^{14} + \alpha^{11}\right)}_{=0}x^5 + \ldots = \alpha^{12} + \alpha^{12}x + \alpha^{8}x^2.$$

Результати обчислень відповідають даним, отриманим за алгоритмом Берлекемпа—Мессі.

Далі за алгоритмом Форні знаходимо значення помилок. Для цього обчислюємо

$$\Lambda'(x) = \alpha^{14} + \alpha^{14}x^2,$$

і складаємо вираз для Y_l:

$$Y_l = \frac{\Omega\left(X_l^{-1}\right)}{\Lambda'\left(X_l^{-1}\right)},$$

де X_l — виявник помилки.

Підставляючи значення $X_7^{-1} = \alpha^{-7} = \alpha^8$, отримуємо $Y_2 = \alpha^{11}$, $Y_5 = \alpha^5$, $Y_7 = \alpha$.

Таким чином, знайдений многочлен помилок $e(x) = \alpha^{11}x^2 + \alpha^5x^5 + \alpha x^7$ відповідає попередньо заданому.

20.5. ВІРОГІДНІСТЬ ПОЯВИ ПОМИЛОК ДЛЯ КОДІВ РІДА–СОЛОМОНА

Ефективність кодів визначається кількістю помилок, які вони можуть виправити, кількістю надлишкової інформації, яку потрібно додавати, а також складністю реалізації кодування і декодування (як апаратною, так і у вигляді програми для комп'ютера).

Коди Ріда—Соломона надзвичайно ефективні для виправлення пакетів помилок, тобто вони є ефективними в каналах з пам'яттю. Також вони добре зарекомендували себе в каналах з великим набором вхідних символів. Особливістю коду Ріда—Соломона є те, що до коду довжиною n можна додати два інформаційні символи, не зменшуючи при цьому мінімальну відстань. Такий розширений код має довжину $n + 2$ і ту саму кількість символів контролю парності, що і початковий код. Вірогідність появи помилки в декодованому символі P_E, можна записати через вірогідність появи помилки в канальному символі p:

$$P_E = \frac{1}{2^m - 1} \sum_{j=t+1}^{2^m-1} j\left(\frac{2^m - 1}{j}\right) p^j (1-p)^{2^m-1-j}. \qquad (75)$$

Тут t — кількість помилкових бітів у символі, які може виправити код, а символи містять m бітів кожний.

Для деяких типів модуляції вірогідність бітової помилки можна обмежити зверху вірогідністю символьної помилки. Для модуляції MFSK з $M = 2^m$ зв'язок між вірогідністю бітової помилки P_B і вірогідністю появи помилки в декодованому символі P_E задається такою формулою:

$$\frac{P_B}{P_E} = \frac{2^{m-1}}{2^m - 1}. \qquad (76)$$

На рис. 56 наведено залежність P_B від вірогідності появи помилки в канальному символі p, отриману з рівнянь (75) і (76) для різних ортогональних 32-х кодів Ріда—Соломона з можливістю корекції t помилкових бітів у символі та $n = 31$ (тридцять один 5-бітовий символ у кодовому блоці). На рис. 57 подано залежність P_B від $\frac{E_b}{N_0}$ для таких систем кодування у разі використання модуляції MFSK і некогерентної демодуляції в каналі AWGN.

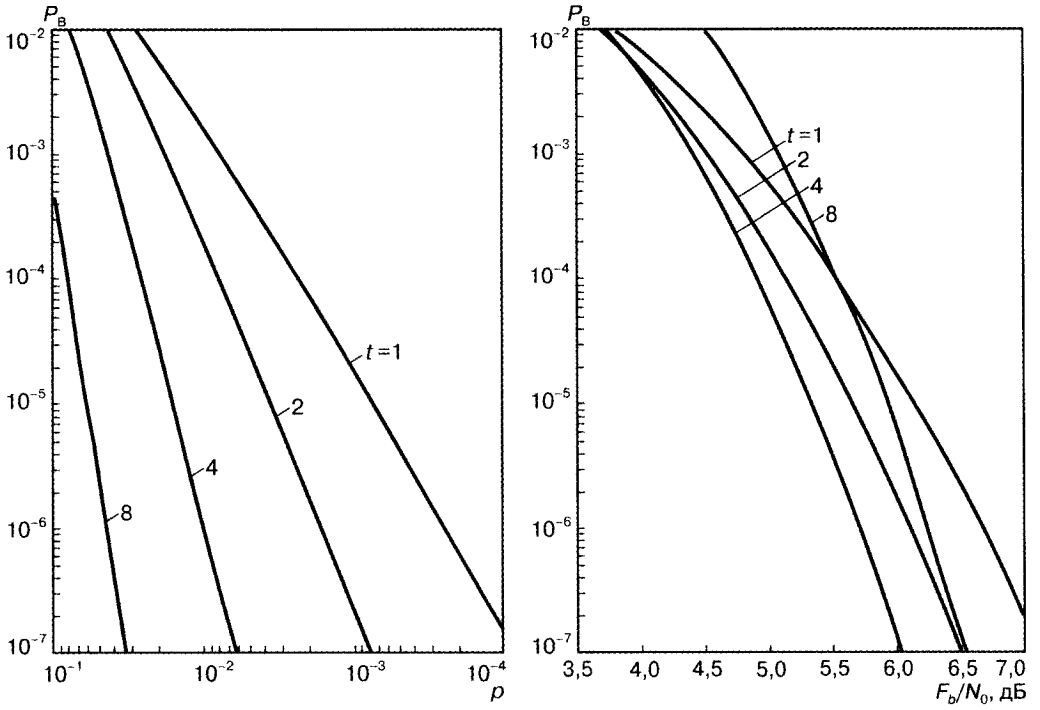


Рис. 56. Залежність $P_в$ від p для різних ортогональних 32-х кодів Ріда—Соломона з можливістю корекції t бітів у символі, $n = 31$

Рис. 57. Залежність $P_в$ від $\frac{E_b}{N_0}$ для різних ортогональних кодів Ріда—Соломона з можливістю корекції t бітів у символі, $n = 31$, при 32-й модуляції MFSK у каналі AWGN

Для кодів Ріда—Соломона вірогідність появи помилок є спадною степеневою функцією довжини блоку n, а складність декодування пропорційна невеликому степеню довжини блоку. Іноді коди Ріда—Соломона застосовуються в каскадних схемах. У таких системах внутрішній згортковий декодер спочатку здійснює деякий захист від помилок унаслідок використання м'якої схеми рішень на виході демодулятора; потім згортковий декодер передає дані, оформлені згідно з жорсткою схемою, на зовнішній декодер Ріда—Соломона, що знижує вірогідність появи помилок. У п. 20.7.3 і 20.7.9 розглянемо каскадне кодування і декодування кода Ріда—Соломона.

20.6. ВИПРАВЛЕННЯ ПОМИЛОК І СТИРАНЬ

Використання стирань спрощує процедуру декодування, оскільки для стертого символу відоме хоча б його місцеположення. Недоліком є розширення алфавіту символів, що приймаються, порівняно з алфавітом переданих.

Якщо кодова відстань коду дорівнює d і прийнята комбінація містить μ помилкових символів, які відповідають стиранням, то у разі декодування μ

стертих символів можна ігнорувати. При цьому дві кодові комбінації відрізнятимуться одна від одної щонайменше в $d - \mu$ інших позиціях, що дає змогу, крім як стирати, виправляти $\left[\frac{d-\mu-1}{2}\right] = \nu$ помилок. Таким чином, код може виправляти всі комбінації з μ стирань і ν помилок у кодовій комбінації, якщо задовольняється умова

$$d > 2\nu + \mu.$$

Складність реалізації виправлення стирань в кодах Ріда—Соломона сумірна зі складністю реалізації виправлення помилок. За аналогією з виправленням помилок вводиться многочлен виявників стирань:

$$\Gamma(x) = \prod_{k=1}^{\mu}(1 - xU_k). \tag{77}$$

Знаючи місця стирань, можна обчислити многочлен $\Gamma(x)$ і підставити на ці місця випадкові значення символів. У гіршому разі це призведе до появи μ додаткових помилок, місцеположення яких відоме.

Ключове рівняння для випадку виправлення стирань і помилок має вигляд

$$S(x) \cdot \Lambda(x) \cdot G(x) = \Omega(x)(\text{mod } x^{2t}). \tag{78}$$

У разі декодування необхідно визначити $\Gamma(x)$ мінімального степеня. Припускаючи, що при передачі кодової комбінації по каналу в ній з'явилося n помилок, а на стертих позиціях додано μ нових помилок, отримуємо

$$[S(x)\Lambda(x)\Gamma(x)]_{\nu+\mu}^{2t-1} = 0. \tag{79}$$

Це означає, що ключове рівняння слід розглядати як систему $2t - \nu - \mu$ лінійних рівнянь для n невідомих коефіцієнтів $\Lambda(x)$. Ця система має розв'язок за $2\nu + \mu \le 2t$.

Визначивши многочлен $\Lambda(x)$, у ході розв'язування ключового рівняння отримуємо многочлен $\Lambda(x) \cdot G(x) = \Psi(x)$, що є загальним многочленом виявників помилок і стирань — многочлен виявників спотворень і многочлен значень помилок $\Omega(x)$. Потім за процедурою Ченя, рівносильною повному перебору всіх можливих значень помилок, знаходимо положення помилок. Коли многочлен перетворюється на нуль — корені знайдено. Далі за алгоритмом Форні визначається характер помилки. Для цього спочатку шляхом згортки многочлена синдрому многочленом виявника Λ отримуємо деякий проміжний многочлен, умовно позначений Ω. Потім за цим многочленом розраховуємо нульову позицію помилки (zero error location), яка в свою чергу ділиться на похідну від Λ-многочлена. Як наслідок утворюється бітова маска, кожний із визначених бітів якої відповідає спотвореному біту (так звана маска спотворених символів). Ця маска накладається на кодове слово і спотворені символи відновлюються. Після цього відкидаються перевірні символи і одержується відновлене інформаційне слово. Для відновлення кодового слова в початковий стан усі спотворені біти повинні бути інвертовані.

У цьому разі в алгоритмі Берлекемпа—Мессі (див. рис. 53) необхідно зробити такі зміни:

- обчислити $\Gamma(x)$;
- замінити $\Lambda(x)$ на $\Psi(x)$;
- обчислення $\Psi(x)$ почати з кроку $r = \deg[G(x)] = \mu$ за наступних початкових умов: $r = L = \mu$, $B(x) = \Psi(x) = \Gamma(x)$;
- замінити критерій перевірки умови, за якої слід збільшити довжину регістра $2L \le \mu + r - 1$ і значення змінної довжини регістра $L \leftarrow r + \mu - L$;
- замінити перевірку виконання умови завершення ітерацій на $r = n - k$.

Проте така процедура не дає змоги отримати многочлен значень помилок $\Omega(x)$.

У прикладі 11 розглянемо процедуру виправлення помилок і стирань за уточненим алгоритмом Берлекемпа—Мессі, наведеним на рис. 58. Многочлен значень помилок $\Omega(x)$ обчислюється за формулою

$$\Omega(x) = S(x) \cdot \Psi(x).$$

Приклад 11. Нехай за умов прикладу 10 многочлен помилок подається так:

$$e(x) = \alpha^{11}x^{10} + \alpha^{7}x^{3} + \alpha^{1}x^{2} + \alpha^{0}x.$$

Причому символи на позиціях, які відповідають x і x^2, прийнято стертими.

Декодер виконує розрахунки компонентів синдрому

$$S_j = \alpha^{11}(\alpha^{j+1})^{10} + \alpha^{7}(\alpha^{j+1})^{3} + \alpha^{1}(\alpha^{j+1})^{2} + \alpha^{0} \cdot \alpha^{j+1},$$

за $j = 0, 1, ..., 5$.

Синдромний многочлен має наступний вигляд:

$$S(x) = \alpha^{3}x^{5} + \alpha^{0}x^{4} + \alpha^{5}x^{3} + \alpha^{12}x^{2} + \alpha^{13}x + \alpha^{0}.$$

Многочлен виявників стирань задається рівнянням

$$G(x) = (1 + x\alpha)(1 + x\alpha^{2}) = \alpha^{3}x^{2} + \alpha^{5}x + 1.$$

Многочлен виявників спотворень $\Psi(x)$ обчислюємо за уточненим алгоритмом Берлекемпа—Мессі (див. рис. 58). Початкові значення — $\deg[G(x)] = 2$, $r = L = 2$, $\Psi_2(x) = B_2(x) = \Gamma(x)$.

Результати розрахунку подано в табл. 36.

За відомим значенням $\Psi(x)$ визначаємо його корені:

$$X_1^{-1} = \alpha^{-1},\ X_2^{-1} = \alpha^{-2},\ X_3^{-1} = \alpha^{-3},\ X_{10}^{-1} = \alpha^{-10}.$$

Таблиця 36. Результати обчислення многочлена виявників

r	S_r	D_r	$M(x)$	$B(x)$	$\Psi(x)$	L
1	α^{0}				1	
2	α^{13}			$\alpha^{3}x^{2} + \alpha^{5}x + 1$	$\alpha^{3}x^{2} + \alpha^{5}x + 1$	2
3	α^{12}	α^{12}	$x^{3} + \alpha^{6}x^{2} + \alpha^{14}x + 1$	$\alpha^{6}x^{2} + \alpha^{8}x + \alpha^{3}$	$x^{3} + \alpha^{6}x^{2} + \alpha^{14}x + 1$	3
4	α^{5}	α^{9}	$\alpha^{3}x^{2} + \alpha^{5}x + 1$	$\alpha^{6}x^{3} + \alpha^{8}x^{2} + \alpha^{3}x$	$\alpha^{3}x^{2} + \alpha^{5}x + 1$	3
5	α^{0}	α^{10}	$\alpha x^{4} + \alpha^{3}x^{3} + \alpha^{8}x^{2} + \alpha^{5}x + 1$	$\alpha^{8}x^{2} + \alpha^{10}x + \alpha^{5}$	$\alpha x^{4} + \alpha^{3}x^{3} + \alpha^{8}x^{2} + \alpha^{5}x + 1$	4
6	α^{3}	α^{7}	$\alpha x^{4} + \alpha^{14}x^{3} + x^{2} + \alpha^{14}x + 1$	$\alpha^{8}x^{3} + \alpha^{10}x^{2} + \alpha^{5}x$	$\alpha x^{4} + \alpha^{14}x^{3} + \alpha^{0}x^{2} + \alpha^{14}x + 1$	4

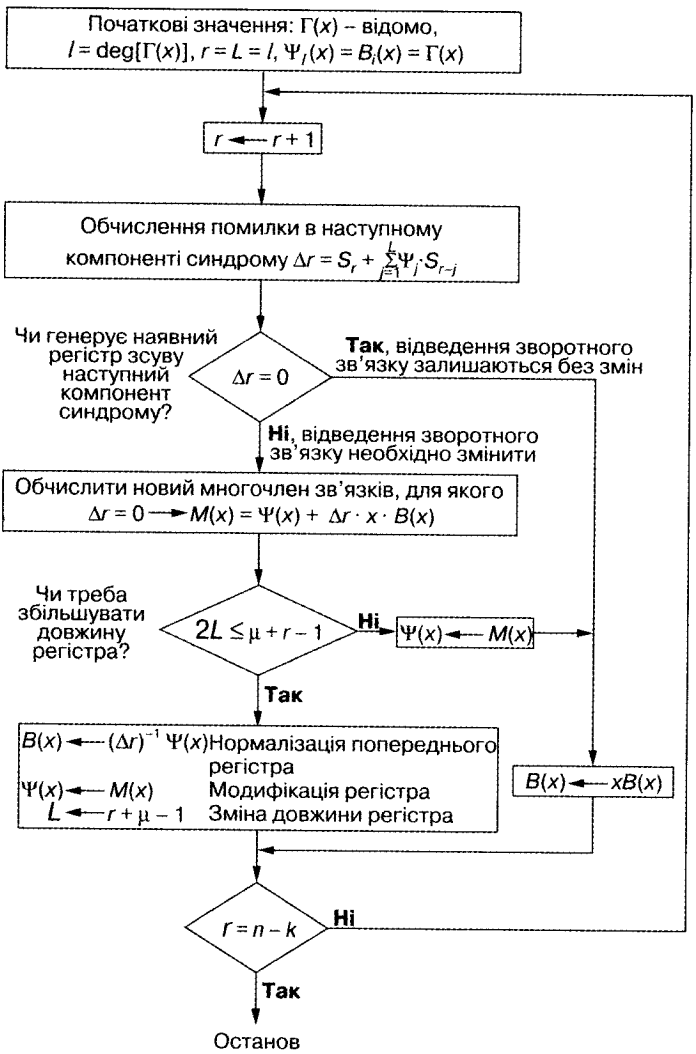


Рис. 58. Блок-схема алгоритму Берлекемпа—Мессі для виправлення стирань та помилок

Для обчислення значень спотворень за формулою $Y_l = \dfrac{\Omega\left(X_l^{-1}\right)}{\Psi'\left(X_l^{-1}\right)}$ знаходимо

$$Y'(x) = \alpha^{14}x^2 + \alpha^{14},$$

$$\Omega(x) = S(x) \cdot \Psi(x) = (\alpha^0 + \alpha^{13}x + \alpha^{12}x^2 + \alpha^5x^3 + \ldots)(\alpha^0 + \alpha^{14}x + \alpha^0x^2 + \alpha^{14}x^3 + \ldots) =$$
$$= \alpha^6x^3 + \alpha^0x^2 + \alpha^2x + \alpha^0.$$

Тоді значення спотворень —

$$Y_1 = \frac{\Omega\left(\alpha^{-1}\right)}{\Psi'\left(\alpha^{-1}\right)} = \frac{\alpha^0 + \alpha + \alpha^{13} + \alpha^3}{\alpha^{14} + \alpha^{12}} = \frac{\alpha^5}{\alpha^5} = \alpha^0,$$

$$Y_2 = \frac{\Omega\left(\alpha^{-2}\right)}{\Psi'\left(\alpha^{-2}\right)} = \frac{\alpha^{11} + \alpha^0}{\alpha^{14} + \alpha^{10}} = \frac{\alpha^{12}}{\alpha^{11}} = \alpha^1,$$

$$Y_3 = \frac{\Omega\left(\alpha^{-3}\right)}{\Psi'\left(\alpha^{-3}\right)} = \frac{\alpha^0 + \alpha^{14} + \alpha^9 + \alpha^{12}}{\alpha^{14} + \alpha^8} = \frac{\alpha^{13}}{\alpha^6} = \alpha^7,$$

$$Y_{10} = \frac{\Omega\left(\alpha^{-10}\right)}{\Psi'\left(\alpha^{-10}\right)} = \frac{\alpha^0 + \alpha^7 + \alpha^{10} + \alpha^6}{\alpha^{14} + \alpha^9} = \frac{\alpha^0}{\alpha^4} = \alpha^{11}.$$

Обчислений многочлен помилок має вигляд

$$e(x) = \alpha^{11}x^{10} + \alpha^7x^3 + \alpha^1x^2 + \alpha^0x.$$

20.7. ПРОЦЕДУРА КОДУВАННЯ ДЛЯ КОДІВ РІДА–СОЛОМОНА

У параграфі 20.1 (див. рівняння (40)) наведено найпоширенішу форму запису кодів Ріда—Соломона через параметри n, k, t і деяке додатне число $m > 2$. Запишемо це рівняння ще раз:

$$(n, k) = (2^m - 1, 2^m - 1 - 2t).$$

Тут $n - k = 2t$ — число контрольних символів; t — кількість помилкових бітів у символі, які може виправити код. Многочлен, що генерує для коду Ріда—Соломона, має такий вигляд:

$$g(X) = g_0 + g_1X^2 + g_2X + \ldots + g_{2t-1}X^{2t-1} + g_{2t}X^{2t}. \qquad (80)$$

Степінь поліноміального генератора дорівнює числу контрольованих символів. Коди Ріда—Соломона є підмножиною кодів БЧХ, наведених у розд. 18. Тому зв'язок між степенем поліноміального генератора і числом контрольних символів такий, як і в кодах БЧХ. У цьому можна переконатися, перевіривши будь-який генератор кодів БЧХ. Оскільки поліноміальний генератор має порядок $2t$, то потрібно мати точно $2t$ послідовні степеня α, які є коренями многочлена. Позначимо корені $g(X)$ як α, α^2, ..., α^{2t}. Немає необхідності починати саме з кореня α, це можна зробити за допомогою будь-якого іншого степеня α. Візьмемо, наприклад, код (7, 3) з можливістю виправляти двосимвольні помилки. Виразимо поліноміальний генератор через $2t = n - k = 4$ кореня наступним чином:

$$\begin{aligned}
&g(X) = (X - \alpha)(X - \alpha^2)(X - \alpha^3)(X - \alpha^4) = \\
&= (X^2 - (\alpha + \alpha^2)X + \alpha^3)(X^2 - (\alpha^3 + \alpha^4)X + \alpha^7) = \\
&= (X^2 - \alpha^4X + \alpha^3)(X^2 - \alpha^6X + \alpha^0) = \\
&= X^4 - (\alpha^4 + \alpha^6)X^3 + (\alpha^3 + \alpha^{10} + \alpha^0)X^2 - (\alpha^4 + \alpha^9)X + \alpha^3 = \\
&= X^4 - \alpha^3X^3 + \alpha^0X^2 - \alpha^1X + \alpha^3.
\end{aligned} \qquad (81)$$

20.7.1. Кодування в систематичній формі

Оскільки код Ріда—Соломона — циклічний, то кодування в систематичній формі аналогічне процедурі двійкового кодування.

Многочлен повідомлення $m(X)$ можна зсунути в крайні праві k розряди регістра кодового слова та додати многочлен парності $p(X)$ у крайні ліві $n - k$ розряди. Помножимо $m(X)$ на X^{n-k}, виконуючи операцію алгебри так, щоб $m(X)$ виявився зсунутим вправо на $n - k$ позицій. (Це неодноразово показано на прикладах двійкового кодування.) Далі поділимо $X^{n-k}m(X)$ на поліноміальний генератор $g(X)$, що можна подати так:

$$X^{n-k}m(X) = q(X)g(X) + p(X). \qquad (82)$$

Тут $q(X)$ і $p(X)$ — відповідно частка і залишок від поліноміального ділення. Як і у разі двійкового кодування, залишок буде парним. Рівняння (82) можна записати у вигляді

$$p(X) = X^{n-k}m(X) \text{ за модулем } g(X). \qquad (83)$$

Результуючий многочлен кодового слова $U(X)$ можна тепер подати так:

$$U(X) = p(X) + X^{n-k}m(X). \qquad (84)$$

Продемонструємо кроки дій процесу кодування, які потрібно зробити згідно з рівняннями (83) і (84), закодувавши повідомлення з трьох символів

$$\underbrace{010}_{\alpha^1}\ \underbrace{110}_{\alpha^3}\ \underbrace{111}_{\alpha^5}$$

за допомогою коду (7, 3), генератор якого визначається рівнянням (81). Спочатку помножимо (зміщення вгору) многочлен повідомлення $\alpha^1 + \alpha^3X + \alpha^5X^2$ на $X^{n-k} = X^4$, одержимо $\alpha^1X^4 + \alpha^3X^5 + \alpha^5X^6$. Далі поділимо такий зміщений вгору многочлен повідомлення на поліноміальний генератор з рівняння (81): $\alpha^3 - \alpha^1X + \alpha^0X^2 - \alpha^3X^3 + X^4$. Поліноміальне ділення недвійкових коефіцієнтів — це більш складна процедура, ніж її двійковий аналог, оскільки операції додавання (віднімання) і множення (ділення) виконуються в полі $GF(q^m)$. На розгляд читача залишимо як допоміжну вправу, перевірити, чи дійсно наслідком поліноміального ділення буде поліноміальний залишок (поліном парності):

$$p(X) = \alpha^0 + \alpha^2X + \alpha^4X^2 + \alpha^6X^3.$$

Зауважимо, що, згідно з рівнянням (84), многочлен кодового слова можна записати у вигляді

$$U(X) = \alpha^0 + \alpha^2X + \alpha^4X^2 + \alpha^6X^3 + \alpha^1X^4 + \alpha^3X^5 + \alpha^5X^6.$$

20.7.2. Систематичне кодування за допомогою ($n - k$)-розрядного регістра зсуву

Як показано на рис. 59, кодування послідовності з трьох символів у систематичній формі на підставі коду (7, 3), що визначається генератором $g(X)$ з рівняння (81), вимагає реалізації послідовного регістра зі зворотними зв'язками.

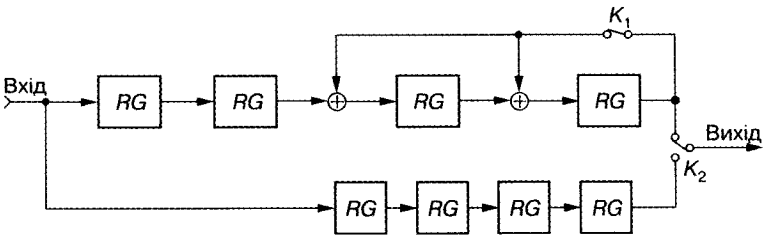


Рис. 59. Схема кодування в систематичній формі

Неважко переконатися, що елементи помножувача на рис. 59, які взято справа наліво, відповідають коефіцієнтам многочлена в рівнянні (81). Цей процес кодування є недвійковим аналогом циклічного кодування, яке описувалося в розд. 18. Тут, відповідно до рівняння (40), ненульові кодові слова утворені $2^m - 1 = 7$ символами і кожен символ складається з $m = 3$ бітів.

Слід зауважити подібність між рис. 59 і рисунками розд. 18. У всіх випадках кількість розрядів у регістрі дорівнює $n - k$. Рисунки в розд. 18 відображають приклад двійкового кодування, де кожен розряд містить 1 біт. У даному розділі наведено приклад двійкового кодування такого, що кожен розряд регістра зсуву (див. рис. 59) містить 3-бітовий символ. На рисунках розд. 18 коефіцієнти, позначені $g_1, g_2, \ldots$, є двійковими, тому вони набувають одного із значень: "0" або "1", просто вказуючи на наявність або відсутність зв'язку в послідовному регістрі. На рис. 59 кожен коефіцієнт є 3-бітовим, тому вони можуть набувати одного з восьми значень.

Недвійкові операції, що здійснюються кодером (див. рис. 59), створюють кодові слова в систематичній формі так само, як і в двійковому випадку. Ці операції визначаються наступними кроками.

Перемикач 1 протягом перших k тактових імпульсів закритий для того, щоб подавати символи повідомлення в $(n - k)$-розрядний регістр зсуву.

Протягом перших k тактових імпульсів перемикач 2 знаходиться в нижньому положенні, що забезпечує одночасну передачу усіх символів повідомлення безпосередньо на регістр виходу (на рис. 59 не показано).

Після передачі k-го символу на регістр виходу перемикач 1 відкривається, а перемикач 2 переходить у верхнє положення.

Інші $n - k$ тактових імпульсів стирають контрольні символи, що містяться в регістрі, подаючи їх на регістр виходу.

Загальна кількість тактових імпульсів дорівнює n, вміст регістра виходу — многочлен кодового слова $p(X) + X^{n-k}m(X)$ ($p(X)$ — кодові символи, $m(X)$ — символи повідомлення в поліноміальній формі).

Для перевірки використовуватимемо ту саму послідовність символів, що і раніше:

$$\underbrace{010}_{\alpha^1}\ \underbrace{110}_{\alpha^3}\ \underbrace{111}_{\alpha^5}.$$

У цьому разі і крайній правий символ, і крайній правий біт також є найпер-

шими. Послідовність дій протягом перших $k = 3$ зсувів у процесі кодування (див. рис. 59) матиме наступний вигляд:

Черга введення			Такт	Вміст регістра				Зворотний зв'язок
α^1	α^2	α^5	0	0	0	0	0	α^5
	α^1	α^2	1	α^1	α^6	α^5	α^1	α^0
		α^1	2	α^2	0	α^2	α^2	α^4
			3	α^0	α^2	α^4	α^6	—

Як бачимо, після третього такту регістр містить чотири контрольних символи α^0, α^2, α^4 і α^6. Потім перемикач 1 переходить у верхнє положення і контрольні символи, що знаходяться в регістрі, подаються на вихід. Тому вихідне слово, записане в поліноміальній формі, можна подати так:

$$U(X) = \sum_{n=0}^{6} u_n X^n ,$$

$$\begin{aligned} U(X) &= \alpha^0 + \alpha^2 X + \alpha^4 X^2 + \alpha^6 X^3 + \alpha^1 X^4 + \alpha^3 X^5 + \alpha^5 X^6 = \\ &= (100) + (001)X + (011)X^2 + (101)X^3 + (010)X^4 + (110)X^5 + (111)X^6. \end{aligned} \qquad (85)$$

Процес перевірки вмісту регістра під час різних тактів дещо складніший, ніж у разі бінарного кодування. Тут додавання і множення елементів поля повинні виконуватися в полі $GF(q^m)$.

Корені поліноміального генератора $g(X)$ мають бути і коренями кодового слова, $g(X)$, що генерується, оскільки правильне кодове слово має вигляд

$$U(X) = m(X)g(X) . \qquad (86)$$

Отже, довільне кодове слово, що виражається через корінь генератора $g(X)$, повинно давати "0". Виникає питання, чи дійсно поліном кодового слова в рівнянні (85) дає "0", коли він виражається через який-небудь з чотирьох коренів $g(X)$. Іншими словами, це означає перевірку рівності

$$U(\alpha) = U(\alpha^2) = U(\alpha^3) = U(\alpha^4) = 0 .$$

Незалежно виконавши обчислення для різних коренів, отримаємо рівняння

$$\begin{aligned} U(\alpha) &= \alpha^0 + \alpha^3 + \alpha^6 + \alpha^9 + \alpha^5 + \alpha^8 + \alpha^{11} = \\ &= \alpha^0 + \alpha^3 + \alpha^6 + \alpha^2 + \alpha^5 + \alpha^1 + \alpha^4 = \\ &= \alpha^1 + \alpha^0 + \alpha^6 + \alpha^4 = \\ &= \alpha^3 + \alpha^3 = 0, \end{aligned}$$

$$\begin{aligned} U(\alpha^2) &= \alpha^0 + \alpha^4 + \alpha^8 + \alpha^{12} + \alpha^9 + \alpha^{13} + \alpha^{17} = \\ &= \alpha^0 + \alpha^4 + \alpha^1 + \alpha^5 + \alpha^2 + \alpha^6 + \alpha^3 = \\ &= \alpha^5 + \alpha^6 + \alpha^0 + \alpha^3 = \\ &= \alpha^1 + \alpha^1 = 0, \end{aligned}$$

$$U(\alpha^3) = \alpha^0 + \alpha^5 + \alpha^{10} + \alpha^{15} + \alpha^{13} + \alpha^{18} + \alpha^{23} =$$
$$= \alpha^0 + \alpha^5 + \alpha^3 + \alpha^1 + \alpha^6 + \alpha^4 + \alpha^2 =$$
$$= \alpha^4 + \alpha^0 + \alpha^3 + \alpha^2 =$$
$$= \alpha^5 + \alpha^5 = 0,$$
$$U(\alpha^4) = \alpha^0 + \alpha^6 + \alpha^{12} + \alpha^{18} + \alpha^{17} + \alpha^{23} + \alpha^{29} =$$
$$= \alpha^0 + \alpha^6 + \alpha^5 + \alpha^4 + \alpha^3 + \alpha^2 + \alpha^1 =$$
$$= \alpha^2 + \alpha^0 + \alpha^5 + \alpha^1 =$$
$$= \alpha^6 + \alpha^6 = 0.$$

Ці обчислення показують, що, як і очікувалося, кодове слово, яке виражається через будь-який корінь генератора $g(X)$, повинно давати "0".

20.7.3. Декодування кодів Ріда–Соломона

У п. 20.1.5 тестове повідомлення кодується в систематичній формі за допомогою коду (7, 3), унаслідок чого маємо многочлен кодового слова, який описується рівнянням (85). Припустимо, що в процесі передачі це кодове слово зазнало спотворення: два символи були прийняті з помилкою. (Така кількість помилок відповідає максимальній здатності коду до виправлення помилок.) У разі використання 7-символьного кодового слова помилкову комбінацію можна подати в поліноміальній формі так:

$$e(X) = \sum_{n=0}^{6} e_n X^n. \tag{87}$$

Нехай двосимвольна помилка є такою, що

$$e(X) = 0 + 0X + 0X^2 + \alpha^2 X^3 + \alpha^5 X^4 + 0X^5 + 0X^6 =$$
$$= (000) + (000)X + (000)X^2 + (001)X^3 + (111)X^4 + (000)X^5 + (000)X^6. \tag{88}$$

Іншими словами, контрольний символ спотворено 1-бітовою помилкою (поданою як α^2), а символ повідомлення — 3-бітовою помилкою (поданою як α^5). У даному випадку прийнятий поліном пошкодженого кодового слова $r(X)$ записується у вигляді суми многочлена переданого кодового слова і многочлена помилкової комбінації, що можна описати рівнянням

$$r(X) = U(X) + e(X). \tag{89}$$

Відповідно до рівняння (89), додавши $U(X)$ до рівняння (85) і $e(X)$ до рівняння (88), одержимо співвідношення

$$r(X) = (100) + (001)X + (011)X^2 + (100)X^3 + (101)X^4 + (110)X^5 + (111)X^6 =$$
$$= \alpha^0 + \alpha^2 X + \alpha^4 X^2 + \alpha^0 X^3 + \alpha^6 X^4 + \alpha^3 X^5 + \alpha^5 X^6. \tag{90}$$

У цьому прикладі з виправлення 2-символьної помилки є чотири невідомих — два з них стосуються розташування помилки, а два — помилкових

значень. Зауважимо важливу відмінність недвійкового декодуванням $r(X)$, яке ми показали в рівнянні (90), від двійкового, яке описувалося в розд. 18. При двійковому декодуванні декодеру потрібно знати лише розташування помилки. Якщо відомо, де знаходиться помилка, біт потрібно поміняти з "1" на "0" або навпаки. Проте тут недвійкові символи вимагають, щоб ми не тільки з'ясували розташування помилки, а й визначили правильне значення символу, розташованого на цій позиції. Оскільки в даному прикладі є чотири невідомих, нам потрібно чотири рівняння, щоб знайти їх.

20.7.4. Обчислення синдрому

Синдром (*syndrome* — грецьке слово, означає сукупність ознак і/або симптомів, що характеризують захворювання) — це результат перевірки парності, що виконується над r, для визначення, чи належить r до набору кодових слів. Якщо r є членом набору кодових слів, то синдром S, що є залишком від ділення кодового слова $c(x)$ на породжувальний многочлен $g(x)$ дорівнює "0", тобто слово залишилося не пошкодженим. Будь-яке ненульове значення S означає наявність помилок. Як і в двійковому випадку, синдром S складається з $n - k$ символів $\{S_i\}$ ($i = 1, \ldots, n - k$). Таким чином, для коду (7, 3) є по чотири символи в кожному векторі синдрому; їх значення можна розрахувати з прийнятого многочлена $r(X)$. Зауважимо, що обчислення полегшуються завдяки самій структурі коду, яка визначається рівнянням (86).

Згідно з цією структурою, кожен правильний многочлен кодового слова $U(X)$ кратний поліноміальному генератору $g(X)$. Отже, корені $g(X)$ також повинні бути коренями $U(X)$. Оскільки $r(X) = U(X) + e(X)$, то $r(X)$, що обчислюється з кожним коренем $g(X)$, має дорівнювати "0", тільки якщо $r(X)$ правильне кодове слово. Тому будь-які помилки призведуть до ненульового результату в одному (або більше) випадку. Обчислення символів синдрому можна подати так:

$$S_i = r(X)|_{X=\alpha^i} = r(\alpha^i), \quad i = 1, \ldots, n-k. \tag{91}$$

Тут, як було закладено в рівнянні (88), $r(X)$ містить два помилкові символи. Якщо $r(X)$ є правильним кодовим словом, то наслідком цього є те, що всі символи синдрому дорівнюватимуть нулю. У наведеному прикладі чотири символи синдрому визначаються таким чином:

$$\begin{aligned} S_1 &= r(\alpha) = \alpha^0 + \alpha^3 + \alpha^6 + \alpha^3 + \alpha^{10} + \alpha^8 + \alpha^{11} = \\ &= \alpha^0 + \alpha^3 + \alpha^6 + \alpha^3 + \alpha^2 + \alpha^1 + \alpha^4 = \alpha^3, \end{aligned} \tag{92}$$

$$\begin{aligned} S_2 &= r(\alpha^2) = \alpha^0 + \alpha^4 + \alpha^8 + \alpha^6 + \alpha^{14} + \alpha^{13} + \alpha^{17} = \\ &= \alpha^0 + \alpha^4 + \alpha^1 + \alpha^6 + \alpha^0 + \alpha^6 + \alpha^3 = \alpha^5, \end{aligned} \tag{93}$$

$$\begin{aligned} S_3 &= r(\alpha^3) = \alpha^0 + \alpha^5 + \alpha^{10} + \alpha^9 + \alpha^{18} + \alpha^{18} + \alpha^{23} = \\ &= \alpha^0 + \alpha^5 + \alpha^3 + \alpha^2 + \alpha^4 + \alpha^4 + \alpha^2 = \alpha^6, \end{aligned} \tag{94}$$

$$\begin{aligned} S_4 &= r(\alpha^4) = \alpha^0 + \alpha^6 + \alpha^{12} + \alpha^{12} + \alpha^{22} + \alpha^{23} + \alpha^{29} = \\ &= \alpha^0 + \alpha^6 + \alpha^5 + \alpha^5 + \alpha^1 + \alpha^2 + \alpha^1 = 0. \end{aligned} \tag{95}$$

Результат підтверджує, що прийняте кодове слово містить помилку (введену нами), оскільки $S \neq 0$.

Приклад 12. Для заданого коду (7, 3) помилкова комбінація відома, оскільки її вибрали попередньо. Звернемося до властивості кодів, коли було введено нормальну матрицю. Всі елементи класу суміжності (рядок) нормальної матриці мають один і той самий синдром. Обчислюючи поліном помилок $e(X)$ із значеннями коренів $g(X)$, потрібно показати, що ця властивість справедлива і для коду Ріда—Соломона. У цьому разі ми повинні одержати ті самі значення синдрому, що і обчислюючи $r(X)$ зі значеннями коренів $g(X)$, тобто ті самі значення, які були отримані в рівняннях (92)—(95).

Розв'язання.

Запишемо

$$S_i = r(X)|_{X=\alpha^i} = r(\alpha^i), \quad i = 1, 2, \ldots, n-k,$$

$$S_i = \left[U(X) + e(X)\right]|_{X=\alpha^i} = U(\alpha^i) + e(\alpha^i),$$

$$S_i = r(\alpha^i) = U(\alpha^i) + e(\alpha^i) = 0 + e(\alpha^i).$$

З рівняння (88) випливає, що $e(X) = \alpha^2 X^3 + \alpha^5 X^4$, тому

$$S_1 = e(\alpha^1) = \alpha^5 + \alpha^9 = \alpha^5 + \alpha^2 = \alpha^3,$$

$$S_2 = e(\alpha^2) = \alpha^8 + \alpha^{13} = \alpha^1 + \alpha^6 = \alpha^5,$$

$$S_3 = e(\alpha^3) = \alpha^{11} + \alpha^{17} = \alpha^4 + \alpha^3 = \alpha^6,$$

$$S_4 = e(\alpha^4) = \alpha^{14} + \alpha^{21} = \alpha^0 + \alpha^0 = 0.$$

На підставі цих результатів можна зробити висновок, що значення синдрому однакові — як отримані шляхом обчислення $e(X)$ із значеннями коренів $g(X)$, так і $r(X)$ з тими самими значеннями коренів $g(X)$.

20.7.5. Локалізація помилки

Припустимо, що в кодовому слові ν помилок, розташованих на позиціях $X^{j_1}, X^{j_2}, \ldots, X^{j_\nu}$. Тоді многочлен помилок, що визначається рівняннями (87), (88), можна записати так:

$$e(X) = e_{j_1} X^{j_1} + e_{j_2} X^{j_2} + \ldots + e_{j_\nu} X^{j_\nu}. \qquad (96)$$

Тут індекси 1, ..., ν позначають помилки від першої до ν-ї включно, індекс j вказує на розташування помилки. Для корегування спотвореного кодового слова потрібно визначити кожне значення помилки e_{jl} і її розташування X^{jl} (l = 1, 2, ..., ν). Позначимо номер виявника помилки $\beta_l = \alpha^{jl}$. Далі обчислюємо $n - k = 2t$ символів синдрому, підставляючи α_i у прийнятий многочлен за i = 1, 2, ..., $2t$. У цьому разі усі $n - k = 2t$ символів синдрому визначаються з наступних

синдромних рівнянь:

$$\begin{aligned} S_1 &= r(\alpha) = e_{j_1}\beta_1 + e_{j_2}\beta_2 + \cdots + e_{j_v}\beta_v, \\ S_2 &= r(\alpha^2) = e_{j_1}\beta_1^2 + e_{j_2}\beta_2^2 + \ldots + e_{j_v}\beta_v^2, \\ &\vdots \\ S_{2t} &= r(\alpha^{2t}) = e_{j_1}\beta_1^{2t} + e_{j_2}\beta_2^{2t} + \cdots + e_{j_v}\beta_v^{2t}. \end{aligned} \tag{97}$$

В цьому випадку є $2t$ невідомих (t значень помилок і t розташувань) і система з $2t$ рівнянь. Втім цю систему з $2t$ рівнянь не можна розв'язати звичайним способом, оскільки рівняння в ній нелінійні (деякі невідомі входять у рівняння в степені, що відрізняється від одиниці). Методика, що дає змогу розв'язати цю систему рівнянь, називається алгоритмом декодування Ріда—Соломона.

Якщо обчислений ненульовий вектор синдрому (один або більше його символів не дорівнюють "0"), то це означає, що прийнято помилку. Далі потрібно дізнатися розташування помилки (або помилок). Многочлен виявника помилок можна визначити так:

$$\begin{aligned} \sigma(X) &= (1+\beta_1 X)(1+\beta_2 X)\ldots(1+\beta_v X) = \\ &= 1+\sigma_1 X + \sigma_2 X^2 + \ldots + \sigma_v X^v \end{aligned} . \tag{98}$$

Корені $\sigma(X)$ — $1/\beta_1$, $1/\beta_2$, ..., $1/\beta_v$. Значення обернені до коренів — номери розташувань помилкової комбінації $e(X)$. Тоді, скориставшись авторегресійною технікою моделювання, складемо з синдромів матрицю, в якій перші t синдромів використовуватимуться для прогнозу наступного синдрому. Матрицю можна записати у вигляді

$$\begin{bmatrix} S_1 & S_2 & S_3 & \cdots & S_{t-1} & S_t \\ S_2 & S_3 & S_4 & \cdots & S_t & S_{t+1} \\ & & & \vdots & & \\ S_{t-1} & S_t & S_{t+1} & \cdots & S_{2t-3} & S_{2t-2} \\ S_t & S_{t+1} & S_{t+2} & \cdots & S_{2t-2} & S_{2t-1} \end{bmatrix} \begin{bmatrix} \sigma_t \\ \sigma_{t-1} \\ \vdots \\ \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} -S_{t+1} \\ -S_{t+s} \\ \vdots \\ -S_{2t-1} \\ -S_{2t} \end{bmatrix}. \tag{99}$$

Скористаємося авторегресійною моделлю рівняння (99), вибравши матрицю найбільшої розмірності з ненульовим визначником. Для коду (7, 3) з корекцією двосимвольних помилок матриця матиме розмірність 2×2. Запишемо цю модель

$$\begin{bmatrix} S_1 & S_2 \\ S_3 & S_4 \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} S_3 \\ S_4 \end{bmatrix}, \tag{100}$$

$$\begin{bmatrix} \alpha^3 & \alpha^5 \\ \alpha^5 & \alpha^6 \end{bmatrix} \begin{bmatrix} \sigma_2 \\ \sigma_1 \end{bmatrix} = \begin{bmatrix} \alpha^6 \\ 0 \end{bmatrix}. \tag{101}$$

Щоб знайти коефіцієнти σ_1 і σ_2 многочлена виявника помилок $\sigma(X)$, спочатку необхідно обчислити обернену матрицю для рівняння (101). У загаль-

ному випадку обернена матриця $[A]^{-1}$ для матриці $[A]$ визначається так:

$$[A]^{-1} = \frac{A_{ij}[A]}{\det[A]},$$

де $A_{ij}[A]$ — алгебраїчне доповнення.

Отже,

$$\det\begin{bmatrix}\alpha^3 & \alpha^5\\ \alpha^5 & \alpha^6\end{bmatrix} = \alpha^3\alpha^6 - \alpha^5\alpha^5 = \alpha^9 + \alpha^{10} = \alpha^2 + \alpha^3 = \alpha^5, \tag{102}$$

$$A_{ij}\begin{bmatrix}\alpha^3 & \alpha^5\\ \alpha^5 & \alpha^6\end{bmatrix} = \begin{bmatrix}\alpha^6 & \alpha^5\\ \alpha^5 & \alpha^3\end{bmatrix}, \tag{103}$$

$$\begin{bmatrix}\alpha^3 & \alpha^5\\ \alpha^5 & \alpha^6\end{bmatrix}^{-1} = \frac{\begin{bmatrix}\alpha^6 & \alpha^5\\ \alpha^5 & \alpha^3\end{bmatrix}}{\alpha^5} = \alpha^{-5}\begin{bmatrix}\alpha^6 & \alpha^5\\ \alpha^5 & \alpha^3\end{bmatrix} = \alpha^2\begin{bmatrix}\alpha^6 & \alpha^5\\ \alpha^5 & \alpha^3\end{bmatrix} = \begin{bmatrix}\alpha^8 & \alpha^7\\ \alpha^7 & \alpha^5\end{bmatrix} = \begin{bmatrix}\alpha^1 & \alpha^0\\ \alpha^0 & \alpha^5\end{bmatrix}. \tag{104}$$

Перевірка надійності. Якщо обернена матриця обчислена правильно, то добуток початкової і оберненої матриць повинен бути одиничною матрицею типу

$$\begin{bmatrix}\alpha^3 & \alpha^5\\ \alpha^5 & \alpha^6\end{bmatrix}\begin{bmatrix}\alpha^1 & \alpha^0\\ \alpha^0 & \alpha^5\end{bmatrix} = \begin{bmatrix}\alpha^4 + \alpha^5 & \alpha^3 + \alpha^{10}\\ \alpha^6 + \alpha^6 & \alpha^5 + \alpha^{11}\end{bmatrix} = \begin{bmatrix}1 & 0\\ 0 & 1\end{bmatrix}. \tag{105}$$

Використовуючи рівняння (101), пошук положень помилок розпочнемо з обчислення коефіцієнтів многочлена виявника помилок $\sigma(X)$ у такій послідовності. Запишемо матрицю

$$\begin{bmatrix}\sigma_2\\ \sigma_1\end{bmatrix} = \begin{bmatrix}\alpha^1 & \alpha^0\\ \alpha^0 & \alpha^5\end{bmatrix}\begin{bmatrix}\alpha^6\\ 0\end{bmatrix} = \begin{bmatrix}\alpha^7\\ \alpha^6\end{bmatrix} = \begin{bmatrix}\alpha^0\\ \alpha^6\end{bmatrix}. \tag{106}$$

З рівнянь (98) і (106) отримаємо

$$\sigma(X) = \alpha^0 + \sigma_1 X + \sigma_2 X^2 = \alpha^0 + \alpha^6 X + \alpha^0 X^2. \tag{107}$$

Корені $\sigma(X)$ обернені до положень помилок. Коли ці корені знайдено, то відомо і розташування помилок. Взагалі, корені $\sigma(X)$ можуть бути одним або декількома елементами поля. Визначимо ці корені шляхом повної перевірки многочлена $\sigma(X)$ зі всіма елементами поля, як буде показано нижче. Будь-який елемент X, для якого $\sigma(X) = 0$, є коренем, що дає нам змогу визначити розташування помилки з використанням таких нерівностей (для коренів рівностей):

$$\sigma(\alpha^0) = \alpha^0 + \alpha^6 + \alpha^0 = \alpha^6 \neq 0,$$
$$\sigma(\alpha^1) = \alpha^2 + \alpha^7 + \alpha^0 = \alpha^2 \neq 0,$$
$$\sigma(\alpha^2) = \alpha^4 + \alpha^8 + \alpha^0 = \alpha^6 \neq 0,$$
$$\sigma(\alpha^3) = \alpha^6 + \alpha^9 + \alpha^0 = 0 \Rightarrow \text{помилка},$$
$$\sigma(\alpha^4) = \alpha^8 + \alpha^{10} + \alpha^0 = 0 \Rightarrow \text{помилка},$$

$$\sigma(\alpha^5) = \alpha^{10} + \alpha^{11} + \alpha^0 = \alpha^2 \neq 0,$$
$$\sigma(\alpha^6) = \alpha^{12} + \alpha^{12} + \alpha^0 = \alpha^0 \neq 0.$$

Як бачимо з рівняння (98), розташування помилок є оберненою величиною до коренів многочлена. Отже, $\sigma(\alpha^3) = 0$ означає, що один корінь маємо за $1/\beta_l = \alpha^3$. Звідси $\beta_l = 1/\alpha^3 = \alpha^4$. Аналогічно $\sigma(\alpha^4) = 0$ означає, що інший корінь з'являється за $1/\beta_{l'} = 1/\alpha^4 = \alpha^3$ (для даного прикладу l і l' позначають першу і другу помилки). Оскільки маємо справу з двосимвольними помилками, многочлен помилок можна записати так:

$$e(X) = e_{j_1} X^{j_1} + e_{j_2} X^{j_2}. \tag{108}$$

У цьому випадку знайдено дві помилки на позиціях α^3 і α^4. Зауважимо, що індексація номерів розташування помилок є суто довільною. Отже, в даному прикладі величини $\beta_l = \alpha^{j_l}$ позначили як $\beta_1 = \alpha^{j_1} = \alpha^3$ і $\beta_2 = \alpha^{j_2} = \alpha^4$.

Слід пам'ятати, що в усіх відомих методах виправлення помилок скоректований код є вірним лише з певною скінченною вірогідністю.

20.7.6. Значення помилок

Раніше помилки позначили e_{jl} (індекс j означає розташування помилки, а індекс l — l-ту помилку). Оскільки кожне значення помилки пов'язане з конкретним місцеположенням, систему позначень можна спростити, позначивши e_{jl} як e_l. Тепер, щоб знайти значення помилок e_1 і e_2, пов'язаних з позиціями $\beta_1 = \alpha^3$ і $\beta_2 = \alpha^4$, можна використовувати будь-яке з чотирьох синдромних рівнянь. Виразимо зі співвідношення (97) рівняння для синдромів S_1 і S_2:

$$\begin{aligned} S_1 &= r(\alpha) = e_1\beta_1 + e_2\beta_2, \\ S_2 &= r(\alpha^2) = e_1\beta_1^2 + e_2\beta_2^2. \end{aligned} \tag{109}$$

Ці рівняння можна записати в матричній формі:

$$\begin{bmatrix} \beta_1 & \beta_2 \\ \beta_1^2 & \beta_2^2 \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} S_1 \\ S_2 \end{bmatrix}, \tag{110}$$

$$\begin{bmatrix} \alpha^3 & \alpha^4 \\ \alpha^6 & \alpha^8 \end{bmatrix} \begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \alpha^3 \\ \alpha^5 \end{bmatrix}. \tag{111}$$

Щоб визначити значення помилок e_1 і e_2, потрібно, як і раніше, знайти обернену матрицю для рівняння (111). Вона визначається так:

$$\begin{bmatrix} \alpha^3 & \alpha^4 \\ \alpha^6 & \alpha^1 \end{bmatrix}^{-1} = \frac{\begin{bmatrix} \alpha^1 & \alpha^4 \\ \alpha^6 & \alpha^3 \end{bmatrix}}{\alpha^3\alpha^1 - \alpha^6\alpha^4} =$$

$$= \frac{\begin{bmatrix} \alpha^1 & \alpha^4 \\ \alpha^6 & \alpha^3 \end{bmatrix}}{\alpha^4 + \alpha^3} = \alpha^{-6} \begin{bmatrix} \alpha^1 & \alpha^4 \\ \alpha^6 & \alpha^3 \end{bmatrix} = \alpha^1 \begin{bmatrix} \alpha^1 & \alpha^4 \\ \alpha^6 & \alpha^3 \end{bmatrix} = \begin{bmatrix} \alpha^2 & \alpha^5 \\ \alpha^7 & \alpha^4 \end{bmatrix} = \begin{bmatrix} \alpha^2 & \alpha^5 \\ \alpha^0 & \alpha^4 \end{bmatrix}. \quad (112)$$

Тепер з матричного рівняння (111) можна розрахувати значення помилок шляхом побудови матриці

$$\begin{bmatrix} e_1 \\ e_2 \end{bmatrix} = \begin{bmatrix} \alpha^2 & \alpha^5 \\ \alpha^0 & \alpha^4 \end{bmatrix} \begin{bmatrix} \alpha^3 \\ \alpha^5 \end{bmatrix} = \begin{bmatrix} \alpha^5 + \alpha^{10} \\ \alpha^3 + \alpha^9 \end{bmatrix} = \begin{bmatrix} \alpha^5 + \alpha^3 \\ \alpha^3 + \alpha^2 \end{bmatrix} \begin{bmatrix} \alpha^2 \\ \alpha^5 \end{bmatrix}. \quad (113)$$

20.7.7. Виправлення прийнятого многочлена за допомогою відомого многочлена помилок

З рівнянь (108) і (113) знаходимо многочлен помилок

$$\hat{e}(X) = e_1 X^{j_1} + e_2 X^{j_2} = \alpha^2 X^3 + \alpha^5 X^4 . \quad (114)$$

Наведений алгоритм відновлює прийнятий многочлен, видаючи як наслідок передбачуване передане кодове слово і, зрештою, декодоване повідомлення. При цьому для многочлена переданого кодового слова і прийнятого многочлена пошкодженого кодового слова маємо відповідно такі вирази:

$$\hat{U}(X) = r(X) + \hat{e}(X) = U(X) + e(X) + \hat{e}(X) , \quad (115)$$

$$\begin{aligned} r(X) &= (100) + (001)X + (011)X^2 + (100)X^3 + (101)X^4 + (110)X^6 + (111)X^6, \\ \hat{e}(X) &= (000) + (000)X + (000)X^2 + (001)X^3 + (111)X^4 + (000)X^5 + (000)X^6, \\ \hat{U}(X) &= (100) + (001)X + (011)X^2 + (101)X^3 + (010)X^4 + (110)X^5 + (111)X^6 = \\ &= \alpha^0 + \alpha^2 X + \alpha^4 X^2 + \alpha^6 X^3 + \alpha^1 X^4 + \alpha^3 X^5 + \alpha^5 X^6. \end{aligned} \quad (116)$$

Оскільки символи повідомлення містяться в крайніх правих $k = 3$ символах, декодованим буде повідомлення

$$\underbrace{010}_{\alpha^4} \ \underbrace{110}_{\alpha^3} \ \underbrace{111}_{\alpha^5} .$$

Це повідомлення точно відповідає тому, яке було вибрано для такого прикладу в п. 20.1.5.

20.7.8. Коди з чергуванням (перемежуванням) і каскадні коди

Переваги різних способів кодування можна об'єднати, застосувавши ***каскадне кодування***. У цьому разі інформація спочатку кодується одним кодом, а потім іншим, як наслідок одержують код-добуток.

Наприклад, популярною є наступна конструкція: дані кодуються кодом Ріда—Соломона, потім перемежуються (при цьому символи, що розташовані

близько, поміщаються далеко один від одного) і кодуються згортковим кодом. На приймачі спочатку декодується згортковий код, далі здійснюється зворотне перемежування (при цьому пакети помилок на виході згорткового декодера потрапляють в різні кодові слова коду Ріда—Соломона), а потім декодування коду Ріда—Соломона.

У попередніх розділах вважали, що у каналі відсутня пам'ять, оскільки розглядалися коди, які повинні протистояти випадковим незалежним помилкам. Канал з пам'яттю — це такий канал, в якому виявляється взаємна залежність погіршень передачі сигналу. Канал, в якому проявляється завмирання унаслідок багатопроменевого поширення, коли сигнал надходить на приймач по двох або більше шляхах різної довжини, — приклад каналу з пам'яттю. Наслідком цього є різна фаза сигналів, тому сумарний сигнал виявляється спотвореним. Такий ефект притаманний каналам мобільного безпровідного зв'язку, а також іоносферним і тропосферним каналам. У деяких каналах також є комутаційні та інші імпульсні перешкоди (наприклад, телефонні канали або канали зі створюваними імпульсними перешкодами). Усі ці перешкоди корелюють у часі і, як наслідок, дають статистичну взаємну залежність успішно переданих сигналів. Іншими словами, спотворення викликають помилки, що мають вид пакетів, а не окремих ізольованих помилок.

Зазвичай, під час передачі інформації по каналу зв'язку вірогідність помилки залежить від відношення сигнал/шум на вході демодулятора, тому за постійного рівня шуму вирішальне значення має потужність передавача. У системах супутникового, мобільного, а також інших типів зв'язку гостро стоїть питання економії енергії. Крім того, в певних системах зв'язку (наприклад, телефонного) суттєво підвищувати потужність сигналу не можна через технічні обмеження.

Оскільки завадостійке кодування дає змогу виправляти помилки, то застосовуючи його, можна понизити потужність передавача, залишаючи швидкість передачі інформації незмінною. Енергетичний виграш визначається як різниця відношення сигнал/шум за наявності та відсутності кодування.

Якщо канал має пам'ять, то помилки не є незалежними, одиничними і випадково розподіленими. Більшість блокових і згорткових кодів розробляється для боротьби з незалежними випадковими помилками. Вплив каналу з пам'яттю на кодований таким чином сигнал призведе до погіршення достовірності передачі. Існують схеми кодування для каналів з пам'яттю, але найбільша проблема в цьому кодуванні — розрахунок точних моделей сильно нестаціонарних статистик таких каналів. Один підхід, за якого потрібно знати тільки обсяг пам'яті каналу, а не його точний статистичний опис, використовує часове рознесення, або чергування бітів.

Чергування бітів кодованого повідомлення перед передачею і зворотна операція після прийому приводять до розсіювання пакета помилок у часі: таким чином, для декодера вони є випадково розподіленими. У реальній ситуації пам'ять каналу зменшується з часовим розділенням, тому ідея, на якій ґрунтується метод чергування бітів, полягає в рознесенні символів кодових слів у часі. Отримувані проміжки часу так само заповнюються символами інших кодових слів. Рознесення символів у часі ефективно перетворює канал

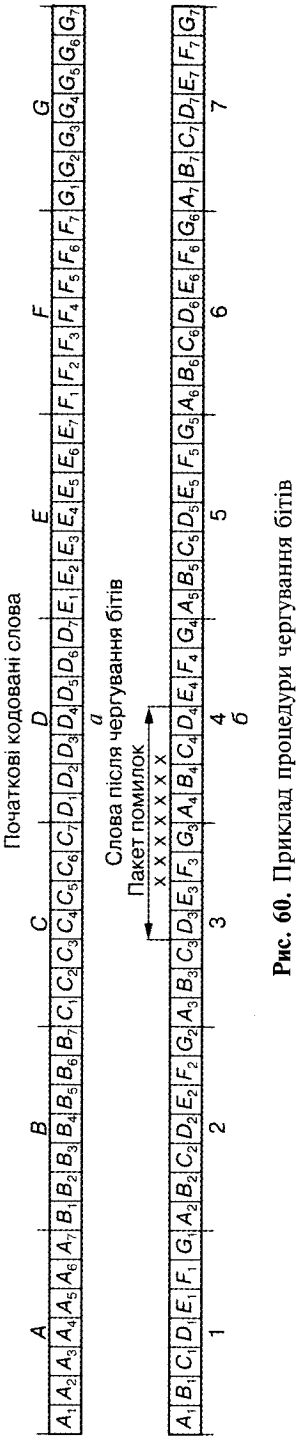


Рис. 60. Приклад процедури чергування бітів

з пам'яттю на канал без пам'яті і, отже, дає змогу використовувати коди з корегуванням випадкових помилок у каналі з імпульсними перешкодами.

Метод чергування (перемежування) байт запропоновано і реалізовано на фірмі IBM для компенсації слабкості корегувального алгоритму під час запису інформації на магнітний носій. Відомо, що дефекти магнітної поверхні мають тенденцію утворювати не одиничні, а групові помилки. Вінчестер IBM 3370 мав чергування (перемежування) 3:1, тобто спочатку йшов перший байт першого блоку, далі — перший байт другого блоку, за ним — перший байт третього блоку, і тільки потім у такій самій послідовності повторювалися інші байти цих блоків. Така процедура підсилювала корегувальну здатність вінчестера з однієї одиничної помилки до трьох ***послідовно*** спотворених байтів. Проте якщо руйнуванню піддавалися не сусідні байти, то корегувальна здатність знову знижувалась до одного спотвореного байта на блок, але вірогідність такої події була незрівнянно меншою.

Варіюючи розмір блоків і ступінь чергування, можна забезпечити кращу або гіршу захищеність за більшої або меншої надлишковості інформації. Дійсно, нехай у нас на диску є N секторів. Тоді, розбивши їх на блоки по 174 сектори в кожному і виділивши три сектори для зберігання контрольної суми, можна відновити щонайменше $N/174$ секторів диска. Виходячи з середньої місткості диска в 100 Гб (що відповідає 209 715 200 секторам), можна відновити до 1 205 259 секторів навіть у разі їх повного фізичного руйнування, витративши всього лише 2 % дискового простору для зберігання контрольних сум.

Пристрій чергування змішує кодові символи в проміжку, що становить декілька довжин блоків (для блокових кодів) або декілька довжин кодового обмеження (для згорткових кодів). Необхідний проміжок визначається тривалістю пакета. Подробиці структури бітового перерозподілу повинні бути відомі приймачу, щоб можна було відновити порядок бітів перед декодуванням. На рис. 60 наведено простий приклад чергування бітів. На рис. 60, *а* подано кодові слова, до яких ще не застосовувалася описана операція: від *A* до *G*. Кожне кодове слово складається з семи кодових символів. Нехай код може виправляти однобітові помилки в будь-якій 7-символьній послідовності. Якщо проміжок пам'яті каналу дорівнює тривалості одного кодового слова,

то такий пакет, тривалістю в сім символів, може знищити інформацію в одному або двох кодових словах. Проте припустимо, що після отримання кодованих даних кодові символи перемішуються, як це продемонстровано на рис. 60,*б*. Іншими словами, кожен кодовий символ кожного кодового слова відділяється від свого сусіда на відстань з семи символьних періодів. Далі отриманий потік перетворюється в модульований сигнал і передається по каналу.

Як бачимо з рис. 60, *б* послідовні канальні пакети шуму потрапляють на сім символьних проміжків, впливаючи на один кодовий символ кожного з семи початкових кодових слів. Під час прийому в потоці спочатку відновлюється початковий порядок бітів так, що він стає подібний на початкову кодовану послідовність, зображену на рис. 60, *а*. Потім потік декодується. Оскільки в кожному кодовому слові можливе виправлення одиничної помилки, імпульсна перешкода ніяким чином не впливає на кінцеву послідовність.

Ідея чергування бітів використовується в усіх блокових і згорткових кодах, розглянутих тут і в попередніх розділах. Кодування здійснюється для блоків даних. Зазвичай застосовуються два типи пристроїв чергування — блокові і згорткові (обидва розглядаються далі).

Широко використовується також псевдовипадкове чергування бітів перед передачею. Це приводить до того, що кластери помилок, внесених під час транспортування, виявляються рознесеними випадковим чином в межах блока даних.

20.7.8.1. Блокове перемежування (чергування)

Блоковий пристрій чергування приймає кодовані символи блоками від кодера, переставляє їх, а потім передає змінені символи на модулятор. Як правило, перестановка блоків завершується заповненням стовпців матриці M рядками і N стовпцями ($M \times N$) кодованої послідовності. Коли матриця повністю заповнюється, символи подаються на модулятор (по одному рядку за раз), а потім передаються по каналу. У приймачі пристрій відновлення виконує зворотні операції: він приймає символи з демодулятора, відновлює початковий порядок бітів і передає їх на декодер. Символи надходять у масив пристрою відновлення по рядках і замінюються стовпцями. На рис. 61, *а* наведено приклад пристрою чергування з $M = 4$ рядками і $N = 6$ стовпцями. Записи в масиві відображають порядок, в якому 24 кодових символи потрапляють у пристрій чергування. Вихідна послідовність, що призначена для передавача, складається з кодових символів, які порядково видалені з масиву, як це показано на рис. 60. Нижче подано найважливіші характеристики такого блокового пристрою.

Пакет, який містить менше ніж N послідовних канальних символів, дає на виході пристрою відновлення початкового порядку символів помилки, рознесені між собою принаймні на M символів.

Пакет з bN помилок, де $b > 1$, дає на виході пристрою відновлення пакет, який містить не менше ніж $[b]$ символьних помилок. Кожний з пакетів помилок відокремлений від іншого не менше ніж на $M - [b]$ символів. Запис $]x[$ означає найменше ціле число, не менше ніж x, а запис $[x]$ — найбільше ціле число x, що не перевищує x.

Періодична послідовність одиничних помилок, розділених N символами, дає на виході пристрою відновлення одиничних пакетів помилок довжиною M.

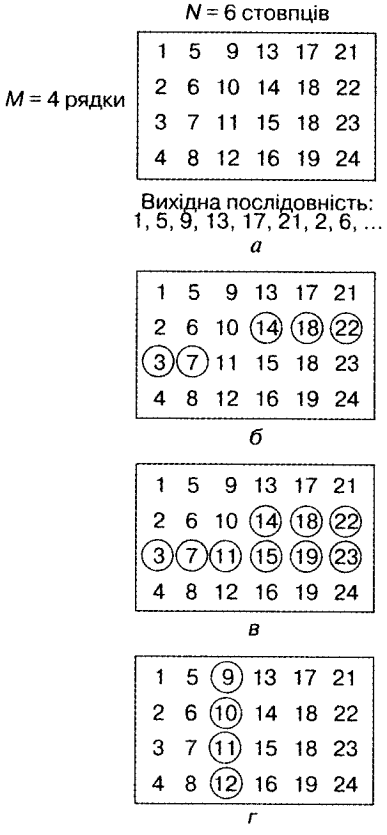


Рис. 61. Приклад пристрою чергування з $M = 4$ рядками і $N = 6$ стовпцями

Пряма затримка між пристроями чергування і відновлення приблизно дорівнює тривалості $2MN$ символів. Якщо бути точним, перш ніж почати передачу, потрібно заповнити лише $M(N - 1) + 1$ елементів пам'яті (як тільки буде внесений перший символ останнього стовпця масиву $M \times X$). Відповідний час потрібний приймачу, щоб почати декодування. Отже, мінімальна пряма затримка — це тривалість $(2MN - 2M + 2)$ символів, не враховуючи затримку на передачу по каналу.

Необхідна пам'ять становить MN символів для кожного об'єкта (пристроїв чергування і відновлення початкового порядку). Проте масив $M \times X$ потрібно заповнити (здебільшого) до того, як його буде зчитано. Для кожного об'єкта потрібно передбачити пам'ять для $2MN$ символів, щоб спорожнити масив $M \times X$, поки інший наповнюватиметься, і навпаки.

Приклад 13. Використовуючи структуру пристрою чергування з $M = 4$, $N = 6$ (див. рис. 61, *а*), перевірте наведені вище характеристики.

Розв'язання

1. Нехай є пакет шуму тривалістю п'ять символьних інтервалів такий, що символи, виділені на рис. 61, *б*, зазнають спотворення під час передачі. Після відновлення початкового порядку бітів у приймачі послідовність набуде вигляду

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

Тут виділені символи є помилковими. Як бачимо, мінімальна відстань, яка розділяє символи з помилками — $M = 4$.

2. Нехай $b = 1{,}5$, тому $bN = 9$. Приклад дев'ятисимвольного пакета помилок наведено на рис. 61, *в*. Після того, як в приймачі відбудеться процедура відновлення початкового порядку бітів, послідовність набуде вигляду

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

Знову виділені символи є помилковими. Тут можна бачити, що пакети містять не більше ніж $[1{,}5] = 2$ символи підряд і рознесені принаймні на $[1{,}5] = 4 - 1 = 3$ символи.

3. На рис. 61, *г* зображено послідовність одиничних помилок, розділених (кожна окремо) $N = 6$ символами. Після відновлення початкового порядку в приймачі послідовність стає такою:

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

Як бачимо, після цього послідовність містить пакет одиничних помилок довжиною $M = 4$ символи.

Пряма затримка — мінімальна пряма затримка, викликана обома пристроями — становить $(2MN - 2M + 2) = 42$ символьних періоди.

Необхідний обсяг пам'яті — розмірність масивів пристроїв чергування і відновлення — становить $M \times X$. Отже, досягається потрібний обсяг пам'яті для зберігання $MN = 24$ символів на обох кінцях каналу. Як уже зазначалося, в загальному випадку пам'ять реалізується для зберігання $2MN = 48$ символів.

Як правило, параметри пристрою чергування, що використовується спільно з кодом з корегуванням одиничних помилок, вибираються так, щоб число стовпців N перевищувало очікувану довжину пакета. Обране число рядків залежить від того, яка схема кодування використовуватиметься. Для блокових кодів M повинна бути більшою, ніж довжина кодового блока; для згорткових кодів M має перевищувати довжину кодового обмеження. Тому пакет довжиною N може спричинити в блоці коду (найбільше) одиничну помилку; аналогічно у разі згорткових кодів у межах однієї довжини кодового обмеження буде не більш ніж одна помилка. Для кодів з корегуванням помилок кратності t вибране N повинно лише перевищувати очікувану довжину пакета, що ділиться на t.

20.7.8.2. Згорткове чергування

Згорткові коди, на відміну від блокових, не ділять інформацію на фрагменти і працюють з нею як з суцільним потоком даних. Отже, метод згортки працює з потоками бітів або символів довільної довжини.

Згорткові коди, як правило, породжуються дискретною лінійною інваріантною у часі системою. Тому, на відміну від більшості блокових кодів, згорткове кодування — дуже проста операція, що не можна сказати про декодування. Кодування загортковим кодом виконується за допомогою регістра зсуву, відводи від якого підсумовуються за модулем 2. Таких сум може бути дві (найчастіше) або більше.

Декодування згорткових кодів, як правило, здійснюється за алгоритмом Вітербі, який намагається відновити передану послідовність згідно з критерієм максимальної правдоподібності.

Згорткові коди ефективно працюють у каналі з білим шумом, але погано справляються з пакетами помилок. Крім того, якщо декодер помиляється, то на його виході завжди виникає пакет помилок.

Згорткові пристрої чергування були запропоновані Рамсі (Ramsey) і Форні (Forney). Схему Форні наведено на рис. 62. Кодові символи послідовно подаються

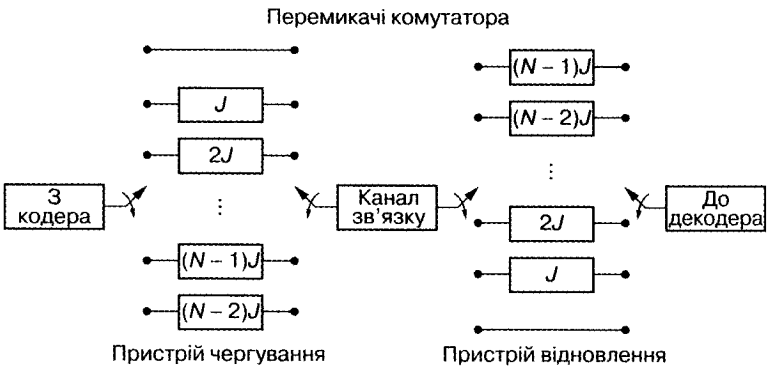


Рис. 62. Структурна схема регістра зсуву для згорткового пристрою чергування/ відновлення

в блок з N регістрів; кожен подальший регістр може зберігати на J символів більше, ніж попередній. Нульовий регістр не призначений для зберігання (символ відразу ж передається). З кожним новим кодовим символом комутатор перемикається на новий регістр, і кодовий символ подається на нього доти, поки "найстаріший" кодовий символ у регістрі не буде переданий на модулятор/передавач. Після $(N-1)$-го регістра комутатор повертається до нульового регістра і процедура повторюється знову. Після прийому операції повторюються в зворотному порядку. І вхід, і вихід пристроїв чергування та відновлення повинні бути синхронізовані.

На рис. 63 наведено приклад простого згорткового чотирирегістрового пристрою чергування, завантаженого послідовністю кодових символів, а також синхронізований пристрій відновлення, який передає оброблені символи на декодер. На рис. 63, *а* показано завантаження символів 1—4; позначка "x" означає невідомий стан. На рис. 63, *б* наведе-

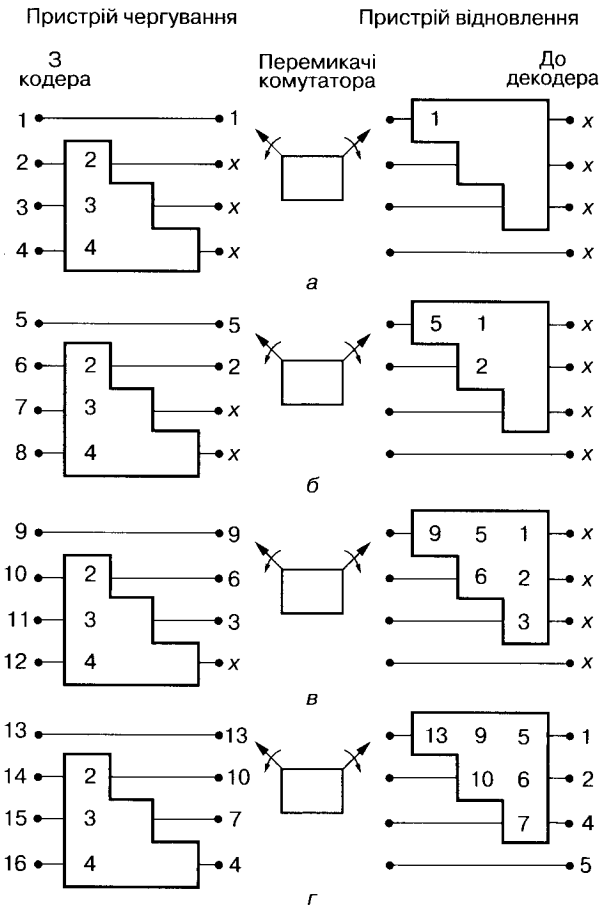


Рис. 63. Приклад згорткового чергування/відновлення

но перші чотири символи, що подаються в регістри, і показано передачу символів 5—8 на вихід пристрою чергування. На рис. 63, *в* зображено символи 9—12, що надходять у пристрій. Тепер пристрій відновлення заповнений символами повідомлення, але ще не здатний нічого передавати на декодер. І нарешті, на рис. 63, *г* показані символи 13—16, що надійшли в пристрій перемішування, а також символи 1—4, передані на декодер. Процес продовжується таким чином доти, поки повна послідовність кодового слова не буде передана на декодер у своїй початковій формі.

Робочі характеристики згорткового пристрою чергування подібні до параметрів блокового пристрою. Важливою перевагою згорткового пристрою над блоковим є те, що при згортковому чергуванні пряма затримка становить $M(N - I)$ символів за $M = NJ$, а необхідні обсяги пам'яті $M(N - 1)/2$ — на обох кінцях каналу. Очевидно, що вимоги до пам'яті і час затримки знижуються удвічі порівняно з блоковим чергуванням.

20.7.9. Каскадні коди

Каскадними називаються коди, в яких кодування здійснюється в два етапи: є внутрішній і зовнішній коди, за допомогою яких і досягається бажана надійність передачі повідомлень. На рис. 64 подано порядок кодування і декодування. Внутрішній код пов'язаний з модулятором. Демодулятор, як правило, настроюється для виправлення більшості канальних помилок. Зовнішній код, найчастіше високошвидкісний (з низькою надлишковістю), знижує вірогідність появи помилок до заданого значення. Основною причиною використання каскадного коду є низький рівень кодування і загальна складність реалізації, менша ніж та, що потрібна для здійснення окремої процедури кодування. На рис. 64 між двома етапами кодування розташовується пристрій чергування. Зазвичай це робиться для того, щоб рознести пакетні помилки, які могли б з'явитися унаслідок внутрішнього кодування.

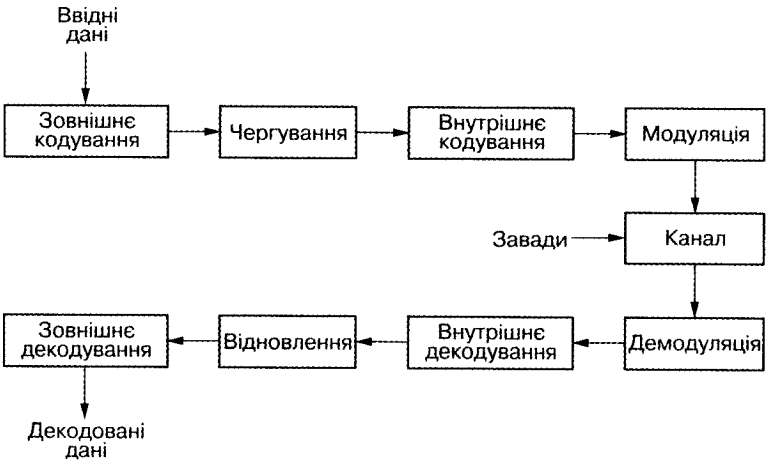


Рис. 64. Блок-схема каскадної системи кодування

У одній з найбільш популярних систем каскадного кодування для внутрішнього коду застосовується згорткове кодування за алгоритмом Вітербі, а для зовнішнього — код Ріда—Соломона з чергуванням між двома етапами кодування. Функціонування таких систем при E_b/N_0, що становить від 0,2 до 2,5 дБ, для досягнення значення $P_B = 10^{-5}$ реально має місце в прикладних задачах. У цій системі демодулятор видає м'яко квантовані кодові символи на внутрішній згортковий декодер, який, у свою чергу, видає жорстко квантовані кодові символи з пакетними помилками на декодер Ріда—Соломона. (У системі декодування за алгоритмом Вітербі вихідні помилки мають тенденцію до появи пакетами.)

Зовнішній код Ріда—Соломона утворюється з m-бітових сегментів двійкового потоку даних. Продуктивність такого (недвійкового) коду Ріда—Соломона залежить тільки від числа символьних помилок у блоці. Код не спотворюється пакетами помилок усередині m-бітового символу. Іншими словами, для даної символьної помилки продуктивність коду Ріда—Соломона така, наче символьна помилка спричинена одним бітом або t бітами. Проте продуктивність каскадних систем дещо погіршується унаслідок корельованих помилок у послідовних символах. Тому перемішування між кодуваннями потрібно виконувати на рівні символів (а не бітів). Каскадні коди спеціально були розроблені для космічної телекомунікації. Далі розглядатимемо поширену практичну реалізацію символьного чергування в каскадних системах.

Процедура кодування ґрунтується на процедурі декодування за припущення, що відомі всі інформаційні символи кодової комбінації, а надлишкові символи стерті, тобто в кодовій комбінації має місце $N - K$ стирань. Для організації процедури кодування надлишковим символам приписуються нульові значення. Процедура кодування будується за алгоритмом швидкого декодування РС-кодів і складається з таких етапів:

1. Обчислення синдромного многочлена.
2. Обчислення многочлена виявників стирань.
3. Обчислення многочлена значень стирань.
4. Обчислення значень надлишкових символів.

Приклад 14. Розглянемо процедуру кодування РС-коду (15, 9) з породжувальним многочленом типу (див. приклад 10)

$$g(x) = x^6 + \alpha^{10}x^5 + \alpha^{14}x^4 + \alpha^4x^3 + \alpha^6x^2 + \alpha^9x + \alpha^6.$$

Нехай інформаційна частина кодової комбінації має вигляд

$$K(x) = \alpha^0x^6 + \alpha^0x^7 + \alpha^0x^8 + \alpha^0x^9 + \alpha^0x^{10} + \alpha^0x^{11} + \alpha^0x^{12} + \alpha^0x^{13} + \alpha^0x^{14}.$$

Тоді двійкове подання $K(x)$ подається так:

0 0 0 0 0 0	α α α α α α α α α	У вигляді степеня
	0 0 0 0 0 0 0 0	примітивного елемента
0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	У вигляді двійкових
0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	послідовностей
0 0 0 0 0 0	0 0 0 0 0 0 0 0 0	
0 0 0 0 0 0	1 1 1 1 1 1 1 1 1	
Надлишкові символи	Інформаційні символи	

Етап 1. Обчислення синдромного многочлена:

$$S_j = \alpha^0(\alpha^{j+1})^6 + \alpha^0(\alpha^{j+1})^7 + \alpha^0(\alpha^{j+1})^8 + \alpha^0(\alpha^{j+1})^9 + \alpha^0(\alpha^{j+1})^{10} + \alpha^0(\alpha^{j+1})^{11} + \alpha^0(\alpha^{j+1})^{12} +$$
$$+ \alpha^0(\alpha^{j+1})^{13} + \alpha^0(\alpha^{j+1})^{14} \text{ за } j = 0, 1, ..., 5,$$
$$S_0 = \alpha^9,\ S_1 = \alpha^3,\ S_2 = \alpha^0,\ S_3 = \alpha^6,\ S_4 = 0,\ S_5 = \alpha^0,$$
$$S(x) = x^5 + \alpha^6 x^3 + x^2 + \alpha^3 x + \alpha^9.$$

Етап 2. Обчислення многочлена виявників стирань (викривлень):

$$\Gamma(x) = Y(x) = (1+x\alpha^0)(1+x\alpha^1)(1+x\alpha^2)(1+x\alpha^3)(1+x\alpha^4)(1+x\alpha^5) =$$
$$= x^6 + \alpha^4 x^5 + \alpha^2 x^4 + \alpha x^3 + \alpha^1 x^2 + \alpha^9 x + 1.$$

Етап 3. Обчислення многочлена значень стирань (викривлень):

$$\Omega(x) = S(x)\cdot\Gamma(x) = (\alpha^9 + \alpha^3 x + x^2 + \alpha^6 x^3 + x^5)(1 + \alpha^9 x + \alpha^{12} x^2 + \alpha x^3 + \alpha^2 x^4 + \alpha^4 x^5 + x^6) =$$
$$= \alpha^4 x^4 + \alpha^1 x^2 + \alpha^9.$$

Тут в $\Omega(x)$ степені, що дорівнюють та старші ніж $N - K = 6$, відкидаються.

Етап 4. Обчислення похідної многочлена виявників стирань:

$$\Gamma'(x) = \alpha^4 x^4 + \alpha^1 x^2 + \alpha^9.$$

Етап 5. Обчислення значень стирань (надлишкових символів):

$$Y_l = \frac{\Omega\left(X_l^{-1}\right)}{\Gamma'\left(X_l^{-1}\right)}$$

за $l = 0, 1, 2, 3, 4, 5$, де $X_l = a_l$ — виявник стирань;

$$Y_0 = Y_1 = Y_2 = Y_3 = Y_4 = Y_5 = 1.$$

Таким чином, комбінація, що кодується, має вигляд

$$f(x) = 1 + x + x^2 + x^3 + x^4 + x^5 + x^6 + x^7 + x^8 + x^9 + x^{10} + x^{11} + x^{12} + x^{13} + x^{14}.$$

20.8. ПРИНЦИПИ, АРХІТЕКТУРА І РЕАЛІЗАЦІЯ КОДІВ РІДА–СОЛОМОНА

20.8.1. Використання кодів Ріда–Соломона

Коди Ріда—Соломона дають можливість виправляти помилки в блоках даних, тому використовуються в цілому ряді додатків, пов'язаних з цифровою комунікацією і зберіганням інформації. Коди Ріда—Соломона використовуються для виправлення помилок у багатьох системах, основними з яких є:

- Запам'ятовувальні пристрої (включаючи стрічку, компакт-диск, DVD, штрих-коди та ін.).
- Безпровідний або мобільний зв'язок (включаючи стільникові телефони, мікрохвильовий зв'язок тощо).
- Супутниковий зв'язок.
- Цифрове телебачення / DVB.
- Високошвидкісні модеми, такі як ADSL, xDSL тощо.

Типова система представлена каналом, зображеним на рис. 65.

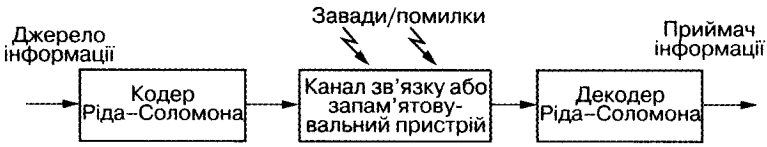


Рис. 65. Структурна схема кодування за допомогою коду Ріда—Соломона

Кодер Ріда—Соломона в цифровий блок даних додає додаткові "надлишкові" біти. Помилки виникають протягом передачі або зберігання інформації з низки причин (наприклад, перешкоди, подряпини на компакт-диску тощо). Декодер Ріда—Соломона обробляє кожен блок даних і намагається виправити помилки, щоб повернути дані до оригінального вигляду. Кількість і тип помилок, які можуть бути виправлені, залежить від характеристик коду Ріда—Соломона.

20.8.2. Властивості кодів Ріда—Соломона

Коди Ріда—Соломона — підмножина кодів БЧХ — лінійні. Такі коди вказуються як коди (n, k) з s бітами символів. Це означає, що кодер до символів даних по s бітів кожний додає символи парності так, щоб ключове слово мало довжину n. Є також $n - k$ символів парності по s бітів кожен. Декодер Ріда—Соломона може виправити аж до t символів, які містять помилки в кодовому слові, де $2t = n - k$.

На рис. 66 наведено діаграму типового коду Ріда—Соломона, відомого як систематичний код: оскільки стан даних зліва незмінний і приєднано символи контролю парності. У разі систематичного кодування в кінець інформаційного блока з k символів приписуються $2t$ перевірних символів, тому операція декодування виконується тільки у випадку дійсного порушення даних. При обчисленні кожного перевірного символу використовуються всі k символів початкового блока. Обчислення систематичних виправляючих кодів Ріда—Соломона здійснюється шляхом ділення інформаційного слова на породжувальний многочлен. У цьому разі всі символи інформаційного слова зсуваються на $(n - k)$ байтів вліво, а на місце, що вивільнилося, записується $2t$ байтів залишку (див. рис. 66).

У цьому випадку при зчитуванні початкового блока ресурси не витрачаються, якщо інформаційне слово не містить помилок, але кодер/декодер повинен виконати $k(n - k)$ операцій додавання і множення для генерації перевірних символів. Крім того, оскільки всі операції виконуються в полі Галуа, то самі операції кодування/декодування вимагають багато ресурсів і часу.

Приклад 15. Популярним є код Ріда—Соломона (255, 223) з 8-бітовими символами. Кожне ключове слово містить 255 байтів кодового слова, з яких 223 байти — дані і 32 — байти парності. Для цього коду $n = 255$, $k = 223$, $s = 8$, $2t = 32$, $t = 16$.

Декодер може виправити будь-які 16 помилок символу в кодовому слові, тобто помилки аж до 16 байтів десь у ключовому слові можуть бути автоматично виправлені.

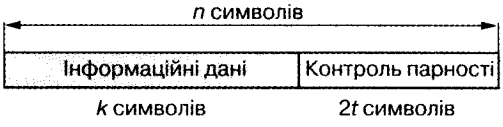


Рис. 66. Діаграма типового коду Ріда—Соломона

Отримаємо розмір символу s для довжини n максимального ключового слова коду Ріда—Соломона: $n = 2^s - 1$. Наприклад, максимальна довжина коду з 8-бітовими символами ($s = 8$) становить 255 байтів.

Коди Ріда—Соломона можна укоротити унаслідок обнулення деякого числа інформаційних символів на вході кодувального пристрою (передавати їх в цьому випадку не потрібно). Під час передачі даних декодеру ці нулі знову вводяться в масив.

Приклад 16. Описаний вище код RC (255, 223) можна скоротити до (200, 168). Кодер вибирає блок 168 байтів даних, додає (концептуально) 55 нульових байтів, створює (255, 223) ключове слово і передає тільки 168 байтів даних і 32 байти парності.

Обсяг обчислювальної потужності, необхідної для кодування і декодування кодів Ріда—Соломона, залежить від числа символів *парності*, які разом з кодом утворюють ключове слово. Велике значення t означає, що можна виправити більшу кількість помилок, але це вимагає більшої обчислювальної потужності порівняно з варіантом за меншого t.

Помилки символу. Одна помилка символу відбувається, коли 1 біт у символі неправильний або коли усі біти в символі неправильні.

Приклад 17. RC (255, 223) може виправити 16 помилок символу. У найгіршому випадку — 16-ти розрядні помилки знаходяться в кожному окремому символі (байті); у кращому — є 16 повних байтових помилок, тоді декодер виправляє 16 × 8 розрядних помилок.

Коди Ріда—Соломона особливо добре виправляють помилки розриву (де серії бітів у ключовому слові отримані з помилкою).

Декодування. Алгебраїчні процедури декодування можуть виправити помилки і стираючі сигнали. Стираючий сигнал формується, коли позиція помилки символу відома. Декодер може виправити аж до t помилок або аж до $2t$ стираючих сигналів. Інформація про стираючі сигнали може часто забезпечуватися демодулятором у цифровій системі комунікації, тобто "прапори" демодулятора отримали ті символи, які, ймовірно, містять помилки.

Коли ключове слово декодоване, то можливими є три результати:

1. Якщо $2s + r < 2t$ (s — помилки, r — стираючі сигнали), тоді оригінальне кодове слово, що передалося, завжди буде відновлено.

Інакше

2. Декодер виявив, що не може повернути оригінальне кодове слово і вказує цей факт.

Або

3. Декодер не виправив виниклі помилки, тоді повертається некоректне кодове слово без будь-якої вказівки.

Вірогідність кожної з трьох можливостей залежить від специфіки коду Ріда—Соломона, а також від кількості і порядку розташування помилок.

Перевага кодування. Перевага використання кодів Ріда—Соломона в тому, що вірогідність помилки, що залишається в декодованих даних (зазвичай), набагато менша, ніж вірогідність помилки, коли код Ріда—Соломона не використовується. Це часто називається виграшем кодування.

Приклад 18. Проектується цифрова телекомунікаційна система, яка працює з BER (Bit Error Ratio), що дорівнює 10^{-9}, тобто з імовірнісним співвідношенням помилки в інформації 10^{-9}, тобто не більше ніж одна помилка в 10^9 бітів інформації. Цього можна досягти підвищенням потужності сигналу відправки інформації або додаючи код Ріда—Соломона (або інший тип того коду, що виправляє помилки). Код Ріда—Соломона дає змогу системі виправляти помилки в інформації з нижчою потужністю сигналу відправника. Енергетична "економія", надана кодом Ріда—Соломона (у децибелах), — перевага кодування.

20.8.3. Архітектура кодування і декодування кодів Ріда—Соломона

Кодування і декодування кодами Ріда—Соломона може здійснюватися або програмним забезпеченням, або в спеціальній апаратній реалізації.

Скінченні (Галуа) арифметичні поля. На кодах Ріда—Соломона базується спеціальний розділ математики, де вивчаються поля Галуа або скінчені поля. Скінчене поле використовує ті самі операції, що і арифметика (додавання, віднімання, множення, ділення та ін.), які впливаючи на елементи поля, формують результат. Для кодера і декодера Ріда—Соломона необхідно виконувати ці самі арифметичні операції. Такі операції вимагають або спеціального устаткування, або програмного забезпечення для здійснення функцій.

Поліноміальний генератор. Кодове слово коду Ріда—Соломона формується за допомогою спеціального многочлена. Усі допустимі кодові слова діляться без залишку на породжувальний многочлен. Загальний аналітичний вигляд генератора многочлена записується так:

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

а кодове слово може бути побудоване за допомогою операції

$$c(x) = g(x) \cdot i(x),$$

де $g(x)$ — многочлен генератора; $i(x)$ — інформаційний блок; $c(x)$ — дійсне кодове слово — примітивний елемент поля.

Прикладом є генератор для РС (255, 249):

$$g(x) = (x - \alpha^{0})(x - \alpha^{1})(x - \alpha^{2})(x - \alpha^{3})(x - \alpha^{4})(x - \alpha^{5}),$$

$$g(x) = x^6 + g_5x^5 + g_4x^4 + g_3x^3 + g_2x^2 + g_1x^1 + g_0.$$

20.8.3.1. Архітектура кодера

Символи парності (кількістю $2t$) у систематичному кодовому слові Ріда—Соломона визначаються за формулою

$$p(x) = i(x) \cdot x^{n-k} \mod g(x).$$

Архітектурно кодер — це сукупність *регістрів зсуву* (shift registers), об'єднаних за допомогою *суматорів* і *помножувачів*, що функціонують за правилами арифметики Галуа. Регістр зсуву — це послідовність комірок пам'яті (*розрядів*),

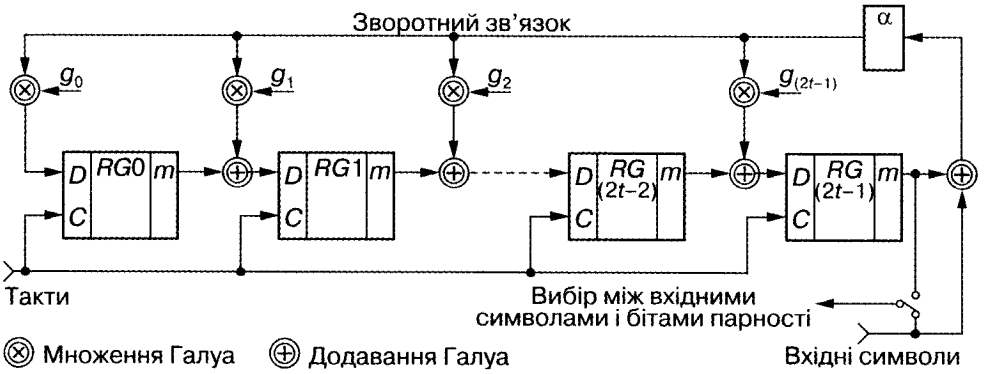


Рис. 67. Спрощена схема кодера Ріда—Соломона

кожний із яких вміщує один елемент поля Галуа $GF(q)$. Дискретно за кожний окремий такт символ, що знаходиться в пам'яті, покидаючи даний розряд, передається на вихідну лінію. Одночасно з цим, комірка пам'яті (розряд) "захоплює" символ, що знаходиться на його вхідній лінії.

У разі апаратурної реалізації регістра зсуву його елементи можуть бути об'єднані як послідовно, так і паралельно. За послідовного об'єднання пересилання одного m-розрядного символу потребує m-тактів, тоді як за паралельного — воно здійснюється всього за один такт.

Низька ефективність програмних реалізацій кодерів Ріда—Соломона пояснюється тим, що розробник не може паралельно об'єднати елементи регістра зсуву, а зобов'язаний здійснювати операції з тією шириною розрядності, яку задає йому архітектура даної машини.

Після n-зсувів на виході регістра з'являється *частка*, а в самому регістрі — *залишок*, які є розрахованими символами парності (вони ж — коди Ріда—Соломона), а коефіцієнти біля членів з g_0 до $g_{(2t-1)}$ чітко відповідають коефіцієнтам породжувального многочлена. Блок-схему кодера Ріда—Соломона для ділення многочлена на константу, зображено на рис. 67.

На рис. 68 наведено архітектуру для систематичного РС (255, 249) кодера.

Кожний з шести регістрів має в собі символ (по 8 біт). Арифметичні оператори здійснюють обмежене доповнення поля або множення на повний символ як на елемент скінченного поля.

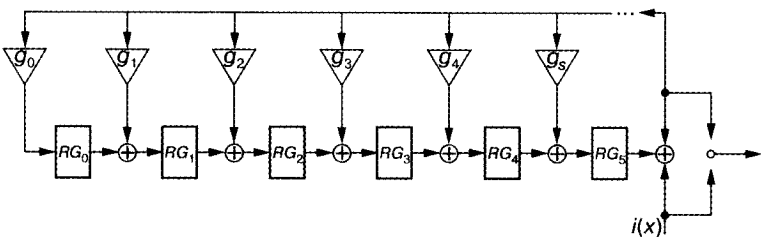


Рис. 68. Схема систематичного кодера РС

20.8.3.2. Архітектура декодера

Декодування кодів Ріда—Соломона — досить складна задача, розв'язування якої — громіздкий, заплутаний і досить ненаочний програмний код. Типова схема декодування, що називається *авторегресійним спектральним методом декодування*, складається із наступних кроків:

- ***Обчислення синдрому помилки*** (синдромний декодер).
- ***Побудова многочлена помилки***, що здійснюється шляхом побудови або високоефективного, але складного в реалізації алгоритму Берлекемпа—Мессі, або простого, але не швидкодійного евклідового алгоритму.
- ***Находження коренів*** даного многочлена, який розв'язується зазвичай простим перебиранням (алгоритм Ченя).
- ***Визначення характеру помилки, що зводиться до побудови*** бітової маски, яка обчислюється за оберненням алгоритму Форні або будь-якого іншого алгоритму обернення матриці.
- ***Виправлення помилкових символів*** накладанням бітової маски на інформаційне слово з наступним інвертуванням усіх помилкових бітів.

Зауважимо, що ця схема декодування не єдина й, імовірно, навіть не найкраща, проте універсальна. Всього існує близько десяти різних схем декодування, абсолютно не подібних одна до одної, вибір яких залежить від того, яка частина декодера реалізується програмно, а яка — апаратно.

Спрощену структурну схему для декодування кодів Ріда—Соломона наведено на рис. 69.

На цій схемі введено такі позначення: $r(x)$ — кодове слово, що надійшло; S_i — синдром; $L(x)$ — поліном виявника помилки; X_i — місце помилок; Y_i — помилка в кількості помилок, що виправляються; $c(x)$ — відновлене кодове слово; ν — число помилок.

Кодове слово $r(x)$, що надійшло на вхід системи, є оригінальним (переданим) словом, а кодове слово $c(x)$ — це оригінальне кодове слово плюс помилки:

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

Декодер Ріда—Соломона здійснює спробу визначити місце і кількість (до t) помилок (або $2t$ стираючих сигналів), а також виправляє помилки або стираючі сигнали.

Розрахунок синдрому. Ці обчислення, аналогічні розрахунку парності. Кодове слово Ріда—Соломона має $2t$ синдромів, які залежать тільки від помилки (кодове слово не передається). Синдроми можуть бути розраховані шляхом заміни $2t$ породжувального многочлена $G(x)$ на корені $R(X)$.

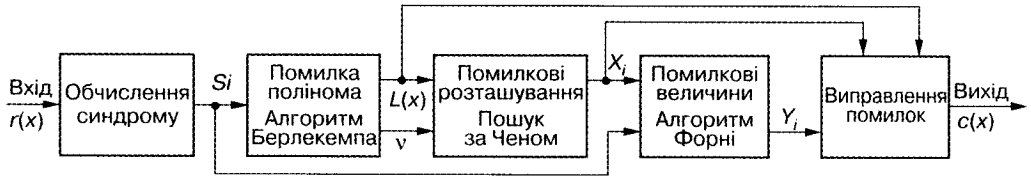


Рис. 69. Схема для декодування кодів Ріда—Соломона

Пошук місця розташування помилкового символу. Цей пошук допускає розв'язування системи рівнянь з t невідомими. Деякі алгоритми можуть зробити це досить швидко. Такі алгоритми використовують спеціальну матричну структуру кодів Ріда—Соломона, що значно скорочує необхідні обчислювальні навантаження. Пошук виконується в два етапи.

1. ***Пошук помилки поліноміальним виявником.*** Такий пошук можна здійснити за допомогою алгоритму Берлекемпа—Мессі або алгоритму Евкліда. Алгоритм Евкліда більш широко використовується на практиці, оскільки його легше реалізувати, проте алгоритм Берлекемпа—Мессі, як правило, приводить до ефективніших апаратних і програмних реалізацій. Коефіцієнти знайденого многочлена безпосередньо відповідають коефіцієнтам помилкових символів в кодовому слові.

2. ***Пошук коренів знайденого полінома.***

• ***Визначення коренів.*** На цьому етапі здійснюється пошук коренів многочлена помилки, які визначають положення спотворених символів у кодовому слові. Реалізується за допомогою процедури Ченя, рівносильної повному перебиранню. У многочлен помилок послідовно підставляються всі можливі значення: якщо многочлен перетворюється на нуль — корені знайдено.

• ***Пошук значення помилки в символі.*** Для цього знову розв'язуємо систему рівнянь з t невідомими. Тут широко використовується алгоритм швидкого пошуку Форні.

20.8.4. Реалізації кодера і декодера Ріда—Соломона

Апаратна реалізація. Існує низка комерційних апаратних реалізацій. Багато наявних систем використовують "готові" вбудовані інтегральні схеми кодування і декодування кодів Ріда—Соломона. Ці мікросхеми налаштовані на підтримку певної мови програмування (наприклад, (255, k), де t від 1 до 16 символів). Останні тенденції пов'язані з реалізацією VHDL або Verilog проектів (логіка ядер або ядер інтелектуальної власності). Такі мікросхеми мають низку переваг над стандартною мікросхемою. Логіка ядра може бути інтегрована з іншими VHDL або Verilog компонентами і синтезована на FPGA (Field Programmable Gate Array) або ASIC (Application Specific Integrated Circuit), що робить можливою реалізацію конструкцій "систем на кристалі", в яких декілька модулів можна об'єднати в одній мікросхемі. Залежно від обсягів виробництва логіка ядра часто за вартістю значно нижча, ніж вартість системи інтегральних схем типу "стандарт". За допомогою логіки ядра проектувальник уникає потенційної необхідності "витрачати свій час на купівлю" інтегральної схеми Ріда—Соломона.

Програмна реалізація. Донедавна впровадження програмного забезпечення в режимі "реального часу" потребувало дуже великої обчислювальної потужності, за винятком простих кодів Ріда—Соломона (тобто кодів за малих значень t). Основна складність реалізації кодів Ріда—Соломона стосовно програмного забезпечення полягає в тому, що загальноцільові процесори не підтримують арифметичних операцій поля Галуа. Наприклад, для реалізації програмного забезпечення поля Галуа потрібний тест на "0", два журнали, що відповідають

таблиці вікна, сума за модулем і антижурнал, що відповідає таблиці. Проте оптимальне складання програм у поєднанні із збільшеною обчислювальною потужністю дають змогу отримувати цілком прийнятні результати щодо високих швидкостей передачі даних для впровадженого програмного забезпечення. У табл. 37 наведено деякі приклади базових показників на 166 МГц Pentium PC. Ці швидкості передач даних мають місце тільки для декодування. Кодування відбувається значно швидше, оскільки вимагає меншої кількості обчислювальних операцій.

Таблиця 37. Значення швидкості передачі даних для деяких кодів на 166 МГц Pentium PC

Код	Швидкість передач даних, Мбіт/с
RC(255, 251)	12
RC(255, 239)	2,7
RC(255, 223)	1,1

20.9. ВПЛИВ ПАРАМЕТРІВ КОДІВ РІДА–СОЛОМОНА НА НАДЛИШКОВІСТЬ

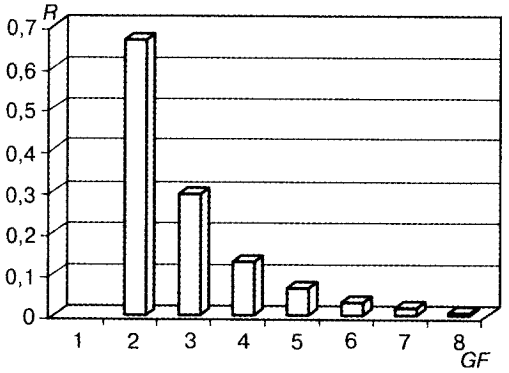
Надлишковість коду і його швидкість залежить, перш за все, від кількості помилок, що виправляються, яка задається при побудові коду. Для зміни надлишковості коду застосовують такі способи:

- зміна параметра b поля Галуа (2^b), на базі якого будується код;
- метод посимвольного перемежування кодів.

Спочатку розглянемо параметри кодів у символах елемента поля Галуа $GF(2^b)$.

Для $s = 1$: $p = 2$, $n = 2b - 1$, $R = p/n = 2/(2b - 1)$. Виправляється один b-бітовий символ. Чим більше b, тим менша надлишковість, отже, більша швидкість коду. Корегувальні можливості і апаратні витрати збільшуються. Залежність надлишковості коду від його параметрів наведено в табл. 38 і на рис. 70.

Для $s = 2$: $p = 4$, $n = 2b - 1$, $R = p/n = 4/(2b - 1)$. Виправляються два b-бітові символи, розташовані в будь-яких двох символах з кодового слова довжиною $2b - 1$. Чим більше b, тим менша надлишковість, отже, більша швидкість коду. Корегувальні можливості збільшуються в Cn^2 разів (в порівнянні з $s = 1$), а апаратні витрати збільшуються незначно, оскільки довжина коду n однакова для $s = 1$ і $s = 2$. Проте швидкість коду в два рази менша, оскільки надлишковість коду R в два рази більша, ніж для $s = 1$.



Розглянемо надлишковість і корегувальні можливості кодів у символах двійкової послідовності b-бітів (застосування кодів Ріда—Соломона для виправлення пакетів помилок). Для випадку з обмеженням характеру розташування помилок отримуємо такий самий результат, як розглянутий раніше для символів елементів поля Галуа (2^b).

Рис. 70. Діаграма зміни параметрів поля Галуа від надлишковості коду

Таблиця 38. Залежність параметрів коду і надлишковості коду за сталого значення $s = 1$

GF	p	n	R
$GF(4)$	2	3	0,67
$GF(8)$	2	7	0,29
$GF(16)$	2	15	0,13
$GF(32)$	2	31	0,065
$GF(64)$	2	63	0,032
$GF(128)$	2	127	0,016
$GF(256)$	2	255	0,008

У разі довільного розташування пакетів помилок: для $s = 1$ довжина пакета $t = 1$, що виправляється (всього 1 біт). Це можна пояснити так. За довжини пакета $t = 2$ в якнайгіршому випадку один спотворений біт може опинитися в одному прийнятому символі кодового слова (b — двійкових символів), а другий — в іншому сусідньому символі, що рівносильно подвійній помилці для кодів Ріда—Соломона. Оскільки код побудовано для виправлення одиничної помилки, то розглянута помилка невиправна. У випадку кодів Ріда—Соломона, що виправляють дві помилки ($s = 2$), довжина довільно розташованого пакета помилок, що виправляється, $t = b + 1$. Це пояснюється тим, що в якнайгіршому випадку за $t = b + 2$ один спотворений біт може опинитися в одному символі кодового слова (b — двійкових символів), b спотворених двійкових бітів — у другому символі, і ще один спотворений біт — у третьому символі. Таким чином, отримали потрійну помилку, яку декодер коду Ріда—Соломона не має змоги виправити, оскільки побудований для коду, що виправляє подвійну помилку. Водночас будь-який пакет помилок спотворених бітів довжиною $t = b + 1$ не зможе розташуватися в трьох сусідніх символах прийнятого кодового слова коду Ріда—Соломона, тому він є виправний.

Таким чином, для одиничного пакета максимальної довжини, який виправляється, збільшення b для $s = 1$ нераціональне; для $s = 2$ збільшення довжини пакета, що виправляється, незначне в порівнянні з методом чергування. Тому застосування кодів Ріда—Соломона, що виправляють одиничні помилки, для виправлення пакетів помилок раціонально тільки у разі їх посимвольного чергування.

Для довільного розташування пакетів помилок максимальної довжини: для $s = 1$ $t = j \cdot b - (b - 1)$; для $s = 2$ $t = 2 \cdot [j \cdot b - (b - 1)]$, де j — параметр чергування. Як бачимо з наведених виразів для залежності довжини пакета помилок, що виправляється, для обох варіантів коду Ріда—Соломона, які допускають синдромне декодування, вона істотно залежить не тільки від параметра чергування j, а і від параметра b поля Галуа $GF(2^b)$.

Надлишковість $R = j \cdot p/j \cdot n = p/n$ не змінюється, отже, швидкість коду також не змінюється, апаратні витрати зростають у j разів.

У разі посимвольного чергування для $s = 1$ з'являються, а для $s = 2$ збільшуються додаткові можливості виправлення множинних пакетів помилок.

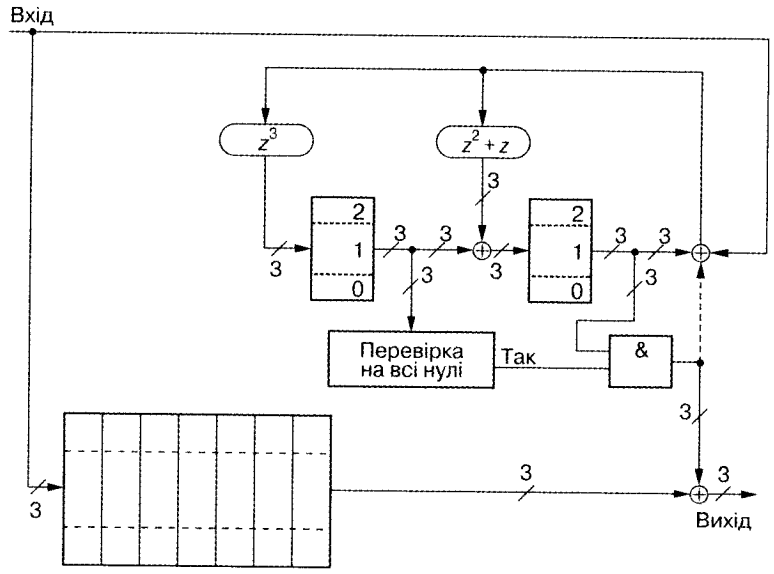


Рис. 71. Декодер коду Ріда—Соломона для поля Галуа $GF(8)$ за $s = 1$

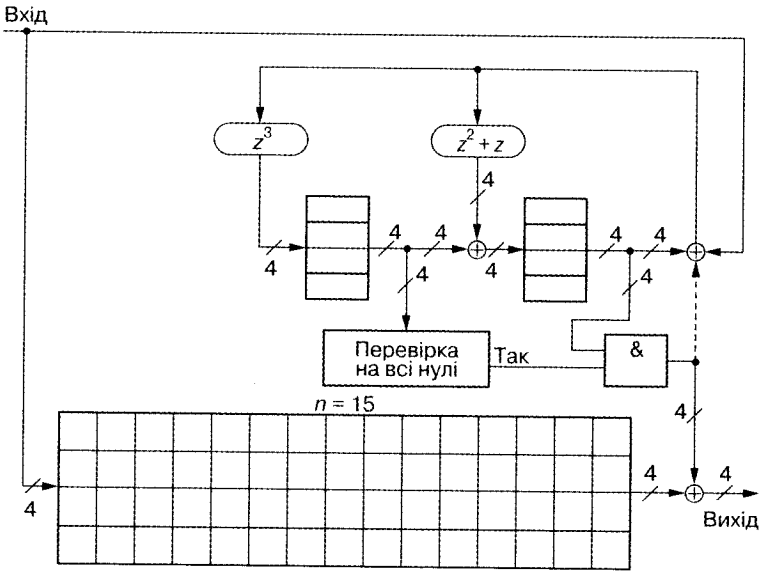


Рис. 72. Декодер коду Ріда—Соломона для поля Галуа $GF(8)$ за $s = 2$

Проте під час побудови кодів слід орієнтуватися на максимальну довжину пакета помилок, що гарантовано виправляються, оскільки в більшості випадків, особливо на носіях інформації (радіальні подряпини на CD-дисках, дефекти в запам'ятовувальних пристроях тощо), помилки згруповано в одиничні пакети. На рис. 71—74 наведено приклади схемної реалізації декодерів кодів Ріда—Соломона з полем Галуа $GF(8)$ і $GF(16)$, що виправляють одиничні або подвійні

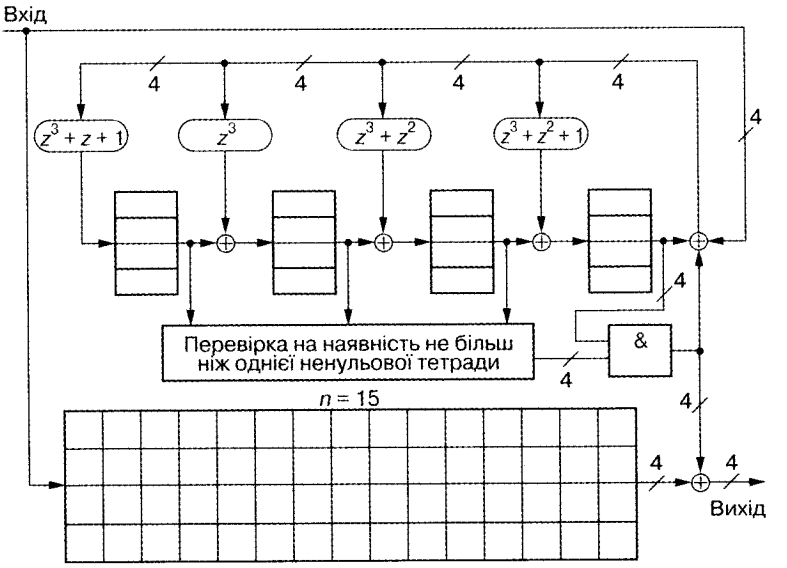


Рис. 73. Декодер коду Ріда—Соломона для поля Галуа $GF(16)$ за $s = 2$

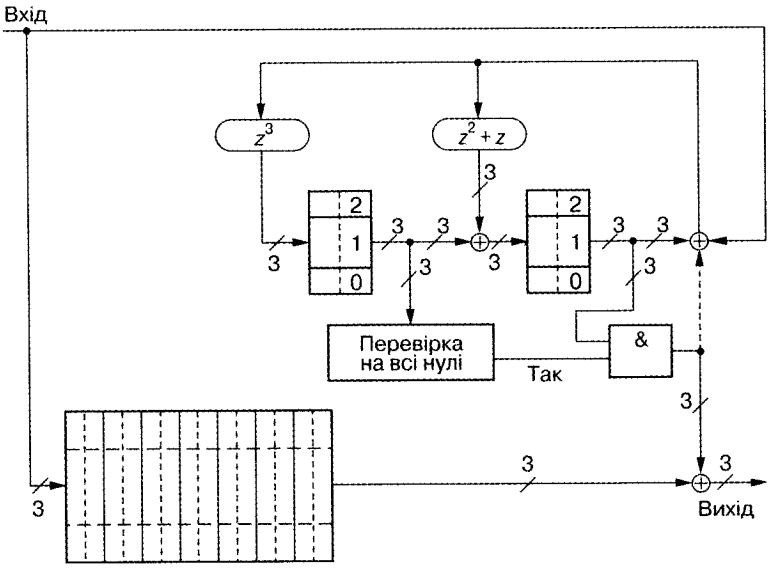


Рис. 74. Декодер коду Ріда—Соломона для поля Галуа $GF(8)$ за $j = 2$

помилки (параметри $s = 1$ і $s = 2$ відповідно), без чергування і з параметром чергування $j = 2$ при $s = 2$.

Кодери для відповідних кодів аналогічні за побудовою декодерам кодів Ріда—Соломона, а також кодерам циклічного коду Хеммінга і кодерам коду БЧХ.

Розділ 21

ЗАВАДОСТІЙКА СИНХРОНІЗАЦІЯ

21.1. СИНХРОНІЗАЦІЯ ПІД ЧАС ПЕРЕДАВАННЯ ДАНИХ

Принципи передавання цифрових сигналів (див. ч. 1) базуються на поняттях "...коди, контроль помилок (їх виявлення та виправлення) та символьна синхронізація".

Передаванню цифрових сигналів передують процеси кодування та модуляції. При переміщенні цифрових сигналів по каналах електрозв'язку вони зазнають різноманітних впливів. На приймальній стороні сам електричний сигнал за допомогою технічних засобів можна з тією або іншою точністю відновити і отримати початкову інформацію. Ефективність технологій демодуляції та декодування цифрових сигналів значною мірою залежить від точного визначення і синхронізації часових інтервалів, насамперед тактових (символьних).

Здатність до синхронізації належить до фундаментальних явищ природи та методів їх дослідження. Щоб точніше зрозуміти зміст поняття синхронізації, варто уявити його біологічні аналоги: це, наприклад, добова синхронізація, яка відчувається унаслідок переїзду з одного часового поясу до іншого, чи синхронізація ритму серця та частоти дихання і руху або соціальні аналоги, наприклад налагоджування "хвиль протесту" або "синхронізація громадської думки" тощо.

Простір і час — ось заходи, які визначають динаміку і статику всього живого та неживого. Ця філософська думка однозначно виділяє синхронізацію як узагальнювальну структуру, яка потрібна в тій чи інший мірі всьому, що належить до всесвіту.

Незважаючи на таку багатозначність, спільне, що визначає поняття синхронізації, — це встановлення та підтримка потрібних часових співвідношень перебігу процесів у часі. Всі наведені явища задовольняють наступні критерії:

- здатність об'єктів генерувати сигнали (ритм серця, рухи, думки тощо);
- наявність каналу "нежорсткого впливу" (механічного, електромагнітного або соціального середовища, впливу через мас-медіа або інтернет);
- здатність захоплювати та відслідковувати ритми (наприклад, оплески, які переходять у ритмічне аплодування).

Дві послідовності називаються синхронними, якщо відповідні події в них відбуваються в один і той самий час. Синхронізація — це процес встановлення і підтримки синхронного стану системи прийому—передачі інформації. У системах передавання інформації в передавачі наявна одна послідовність подій, а в приймачі — інша. У разі передавання дискретних повідомлень сигнали є послідовністю елементів визначеної довжини. Тому необхідною є синхронізація відліків часу в передавачі і в приймачі систем передачі інформації.

Синхронізація цифрових сигналів електрозв'язку (ЦСЕ) — це стан, за якого значущі моменти двох або декількох ЦСЕ перебувають у необхідних фазових співвідношеннях. Значущі моменти ЦСЕ — моменти часу, коли символи ЦСЕ набувають істинного значення з найбільшою ймовірністю.

Тактовий синхросигнал — це спеціальний сигнал, призначений для задання тактових точок інших сигналів. Тактові точки ЦСЕ — це точки на осі часу, що характеризують ідеальне положення значущих моментів ЦСЕ. Тактовий інтервал ЦСЕ — інтервал часу між сусідніми тактовими точками ЦСЕ, який зазвичай називають ***тактом***, а відповідну їм синхронізацію — ***тактовою***. Тактова частота ЦСЕ — число тактових інтервалів ЦСЕ в одиницю часу.

Тактова синхронізація цифрових сигналів електрозв'язку — процес встановлення й підтримки необхідних фазових співвідношень між значущими моментами ЦСЕ й тактовим синхросигналом.

Синхронізація послідовності відліків часу, з якої формуються імпульси опитування даного пристрою, повинна знаходитися в певному фазовому відношенні до одиничних елементів дискретного повідомлення, що приймається. Відстань між імпульсами опитування залежить від способу обробки дискретних повідомлень. Існує два таких способи — обробка в цілому та поелементна обробка повідомлень (стробування). У першому випадку кодова комбінація визначається як складний одиничний сигнал. Якщо кодова комбінація має n одиничних елементів тривалістю τ_0, то опитувані (стробувальні) імпульси повинні відбуватися з інтервалами $n\tau_0$ і збігатися з початком кожної кодової комбінації, що приймається. У іншому випадку опитувані (стробувальні) імпульси мають відбуватися з інтервалом τ_0 і знаходитися в певному фазовому співвідношенні з тими імпульсами, що приймаються одиничним елементом.

У загальному випадку розрізнюють такі цифрові сигнали електрозв'язку:

- ізохронні, для яких інтервал часу між сусідніми тактовими точками (тактовий інтервал) є постійним — такими є всі стандартні цифрові сигнали;
- анізохронні (неізохронні), для яких тактовий інтервал не є постійним (наприклад, символи "1" і "0" кодуються імпульсами різної довжини);
- синхронні, для яких значущі моменти знаходяться в постійному співвідношенні, тобто забезпечується в середньому однакова частота і постійний зсув фази;
- асинхронні — всі, які не є синхронними;
- гомохронні, для яких значущі моменти знаходяться в постійному, але некерованому фазовому співвідношенні;
- мезохронні, для яких значущі моменти з'являються з однаковою середньою швидкістю (два мезохронні сигнали — це ізохронні й асинхронні цифрові сигнали, що мають однакові в середньому частоти і некеровані фазові співвідношення);
- плезіохронні, для яких значущі моменти з'являються з номінально однаковою швидкістю, причому будь-які зміни швидкості обмежені встановленими межами (два плезіохронні сигнали — це ізохронні й асинхронні цифрові сигнали з різними номінальними частотами).

Розрізняють синхронізацію за елементами і циклами. Синхронізація за елементами (саме синхронізація) — процес встановлення відповідності між значущими моментами одиничних елементів при передачі і прийомі. Синхронізація за циклами (фазування за циклами) — процес встановлення відповідності між фіксованими за положенням у кодовій комбінації значущими моментами при передачі і прийомі.

Циклова синхронізація ЦСЕ — це синхронізація ЦСЕ, за якої встановлюються й підтримуються необхідні фазові співвідношення між циклами часового об'єднання переданих і прийнятих ЦСЕ.

Надциклова синхронізація ЦСЕ — це синхронізація ЦСЕ, за якої встановлюються й підтримуються необхідні фазові співвідношення між надциклами часового об'єднання переданих і прийнятих ЦСЕ.

Первинною у разі синхронізації ЦСЕ є тактова синхронізація, тому при досягненні тактової синхронізації за допомогою циклового синхросигналу легко встановити й синхронізацію за циклом.

Після ухвалення рішення щодо кожного елемента кодової комбінації ухвалюється рішення щодо інформаційного символу, який відповідає даній кодовій комбінації. Отже, в приймачі повинні бути сигнали, що вказують початок і кінець кодової комбінації.

Крім наведених, у системах передавання даних на різних рівнях взаємодії розрізняють синхронізацію пакетів або інтернет-телефонії (тобто вирівнювання часових затримок та побудова правильної послідовності надходження пакетів під час передавання інформації сигналів реального часу, наприклад телефонія, відео і т. д.); синхронізацію засобів мультимедіа (встановлення потрібних часових співвідношень при записуванні або демонстрації зображення, тексту, звуку, відео і т. д.).

Спільним для цих різноманітних застосувань є встановлення і підтримка потрібних часових співвідношень перебігу одного або кількох процесів.

Можна сформулювати три основні завдання синхронізації:

- генерування детермінованих періодичних, ізохронних сигналів високої точності або квазіперіодичних сигналів;
- передавання сигналів синхронізації на відстань, у тому числі організація зв'язків між пристроями синхронізації;
- забезпечення захоплення та відстеження синхросигналу.

У технологічному процесі цифрових телекомунікацій термін синхронізація означає, що тактова частота в мережі визначається єдиним джерелом для забезпечення двох умов інформаційного обміну:

- неперервності: середнє значення передавання (прийому) цифрових даних у пов'язаних між собою кінцевих пристроях повинно бути однаковим;
- цілісності: цифрові інформаційні блоки у кінцевому пристрої прийому мають з'являтися з тією самою послідовністю, з якою вони відправляються з кінцевого пристрою передачі.

Таким чином, синхронізація мережі — це узагальнена концепція щодо засобу розподілення спільного для всіх вузлів зв'язку часу або спільної частоти. Сучасні транспортні мережі потребують для розподілення опорних сигналів спеціальної мережі підтримки — мережі синхронізації. У загальному ви-

гляді мережа синхронізації складається з вузлів, з'єднаних каналами синхронізації. Основна задача цифрової мережі полягає в тому, щоб гарантувати отримання однієї і тієї самої швидкості передавання інформації в цифрових мережах. Усі генератори, що встановлені на цифровій мережі, повинні синхронізуватися від одного або кількох генераторів з близькими значеннями частот вихідних коливань.

21.2. ЗАВАДОСТІЙКІСТЬ ДЕМОДУЛЯЦІЇ Й ДЕКОДУВАННЯ ПІД ЧАС ПЕРЕДАВАННЯ ЦИФРОВИХ ДАНИХ

У цифрових системах передавання завадостійкість демодуляції значною мірою забезпечується унаслідок формування в демодуляторі так званих стробувальних (опитувальних) імпульсів (часове положення яких відповідає, наприклад, фронтам тактових імпульсів), що періодично ідуть один за одним і формуються системою тактової синхронізації.

При проходженні елементарних сигналів по реальному каналу завади можуть істотно спотворити форму імпульсу на приймальній стороні. У цьому випадку демодулятор на прийомі буде працювати з погіршеною завадостійкістю. Стійкість роботи демодулятора в загальному випадку буде залежати від параметра зв'язності h дискретного каналу. У разі прийому дискретної послідовності параметр зв'язності повинний перевищувати деяке граничне значення $h_{\text{пор}}$. У цьому випадку реєстрація символів буде здійснюватися без помилок:

$$h^2 \geq h^2_{\text{пор}} = \frac{\overline{P}_{\text{с}}}{P_{\text{ш}}} \cdot \frac{\tau_{\text{і}}}{\xi_{\text{а}}} \cdot F_{\text{еф}}, \qquad (117)$$

де $\overline{P}_{\text{с}}$ — середнє значення потужності елементарного сигналу в місці прийому; $\tau_{\text{і}}$ — тривалість елементарного сигналу (імпульсу); $P_{\text{ш}}$ — потужність адитивних завад (шуму) на вході демодулятора; $F_{\text{еф}}$ — ефективна смуга частот сигналу; $\xi_{\text{а}}$ — коефіцієнт асинхронізму.

Неідеальність тактової синхронізації (наявність коефіцієнта асинхронізму) призводить до зменшення параметра h^2. Аналіз показує, що середньоквадратичне значення неузгодженості фронтів переданого і прийнятого імпульсів, яке становить $\Delta\tau_{\text{і}} \leq 0{,}05\tau_{\text{і}}$, настає вже при значенні коефіцієнта

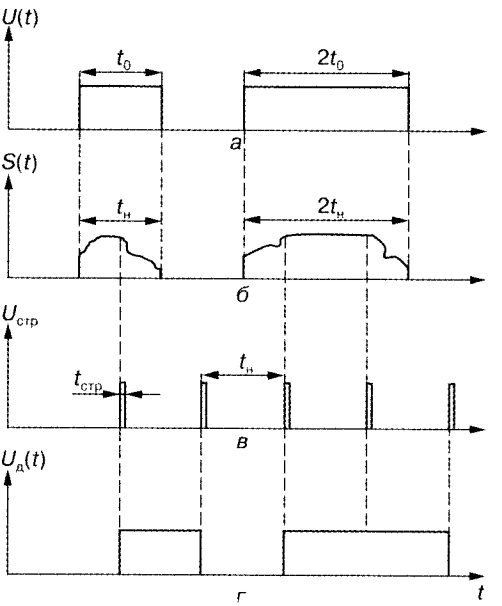


Рис. 74. Демодуляція цифрового сигналу електрозв'язку

асинхронізму ξ_a = 1,3...1,5. Тактова синхронізація забезпечується формуванням у демодуляторі стробувальних імпульсів $U_{стр}$, що виникають один за одним через період $t_н = \tau_i$, причому їх часове положення відповідає значущим моментам прийнятого цифрового сигналу електрозв'язку $S(t)$, отриманого унаслідок переміщення по каналу електрозв'язку переданого сигналу $U(t)$ (рис. 74).

Тривалість стробувальних імпульсів $\tau_{стр}$ значно менша, ніж тривалість інформаційних імпульсів $t_{стр} << \tau_i$. Це дає змогу встановлювати фронти стробувальних імпульсів у моменти максимальних амплітуд обвідної сигналу символів $S(t)$, що надходять з каналу, і приймати рішення стосовно значення сигналу на виході демодулятора $U_д(t)$. За статистикою ці моменти відповідають значущим моментам прийнятого цифрового сигналу електрозв'язку (у нашому випадку середній частині інформаційних імпульсів τ_i). Для проведення такої операції генератор стробувальних імпульсів повинен мати можливість змінювати їх у часі, наприклад, завдяки процесу автоматичного керування в пристрої тактової синхронізації.

21.3. ЗАЛЕЖНІСТЬ ЗАВАДОСТІЙКОСТІ ПЕРЕДАВАННЯ ДАНИХ ВІД СПОТВОРЕННЯ СИНХРОСИГНАЛІВ

Спотворення синхросигналів у цифровому обладнанні виникають унаслідок нестійкої роботи окремих вузлів, дефектів розробки, а також можуть спричинюватися зовнішніми впливами або процесами внутрішньої деградації.

Спотворення тактового синхросигналу може виявлятися як фазове тремтіння (джитер), дрейф фази (вандер) й зміна, яка викликана відхиленням частоти. Крім того, зміна фази може зумовлюватися порушеннями неперервності фази, що викликані перешкодами від перехідних процесів, наприклад, у разі зміни маршрутизації в мережі, а також автоматичним захисним перемиканням. Зміни фази тактового синхросигналу призводять до зниження якості переміщення інформації у вигляді помилок у ЦСЕ та проковзувань ЦСЕ.

Помилки в ЦСЕ — це невідповідність прийнятого символу ЦСЕ переданому. Помилки можуть виникати в точках передавання, прийому, регенерації сигналів унаслідок зсуву тактових синхросигналів від їхніх оптимальних положень у часі. Помилки — основний чинник погіршення якості переміщення інформації, які спричинюють спотворення мовлення в телефонних службах і призводять до втрати або невірогідності інформації чи до зниження пропускної здатності в службах типу передачі даних.

Зниження якості аналогової інформації, переданої за допомогою цифрового кодування, може відбутися через зміни фази імпульсів, що стробують ЦСЕ і відновлюються цифроаналоговими перетворювачами на кінцях тракту переміщення інформації. Це суттєво впливає на відеосигнали, передані за допомогою цифрового кодування.

Проковзування ЦСЕ — це зменшення або збільшення числа тактових інтервалів ЦСЕ, що призводить до втрати інформації через випадання або вставки символів ЦСЕ. Проковзування у вигляді повторення або виключення групи символів у синхронній або плезіохронній послідовності двійкових сигналів виникають через різні швидкості зчитування й запису в буферній пам'яті.

21.3.1. Фазові спотворення синхросигналів

Теоретично функція фазових спотворень $e[kT]$ визначається як різниця між відповідними значущими моментами реального та ідеального (опорного) цифрових сигналів електрозв'язку:

$$e[kT] = t_k - t_k^*, \qquad (118)$$

де k — номер значущого моменту (в загальному випадку належить множині цілих чисел: $k \in Z$, а в окремих випадках — множині натуральних чисел $k \in N$); t_k та t_k^* — k-й значущий момент відповідно ідеального та реального сигналів; T — одиничний інтервал: $T = 1/f_т$ ($f_т$ — тактова частота).

Фазові спотворення розглядаються як паразитна фазова, фазо-імпульсна модуляція (ФІМ) тактової послідовності деяким сигналом завад $\xi(t)$. У термінах ФІМ функція фазових спотворень (86) у кожен момент часу $t_k^* = kT \pm \Delta t_k$ пропорційна значенню модулювального сигналу завад $\xi(t)$. Коефіцієнт пропорційності — амплітуда фазових спотворень (джитеру або вандеру), виражена в абсолютних одиницях часу (Δt_{max}) або нормована відносно величини тактового інтервалу T (індекс модуляції $\beta_{фім}$): $e[kT] = \Delta t_{max}\, \xi(t_k)$, $e[kT] \sim \beta_{фім}\, \xi(t_k)$. Спектральний аналіз показує, що спектр ФІМ сигналу (спектр сигналу з фазовими спотвореннями) містить корисну складову на частоті модуляції Ω: $B_{фім}(\Omega, t) = 2U_0\beta_{фім} \sin(\Omega\tau/2)\, \xi(t)$, яка відображає модулювальний сигнал (функцію фазових спотворень $e[kT]$) і може бути виділена у разі демодуляції ФІМ сигналу за допомогою відповідної фільтрації.

Враховуючи властивості лінійної моделі ФІМ, функцію фазових спотворень (86) у будь-який значущий момент kT можна подати як лінійне перетворення деякого випадкового процесу (послідовності) $\xi(t_k)$:

$$e[kT] = \beta_{фім}\, \xi(t_k), \qquad (119)$$

де $\xi(t_k)$ — значення випадкового фазового зсуву ξ у момент часу kT.

21.3.2. Частотні спотворення синхросигналів

Розходження тактових частот синхросигналів у трактах прийому й передачі приводять до появи проковзувань (slit).

Останні по-різному впливають на якість переміщення інформації різних типів електрозв'язку. Так, у разі передавання мовлення одне проковзування в хвилину (один цикл тривалістю 125 мкс) впливає неістотно (непомітно) на якість прийому мовлення, а у разі передавання даних — неприпустимо. Ефект проковзувань у мовному сигналі, перетвореному в цифрову форму, виявляється зрідка як клацання, тріск, що викликано стрибками фази й амплітуди ЦСЕ. Через високу надлишковість мовних повідомлень вплив проковзувань невеликий. Відчутне клацання в системах з імпульсно-кодовою модуляцією виникає тільки в 1/25 випадках. Вважається, що інтервал часу між суміжними клацаннями, який забезпечує прийнятну якість переміщення мовлення, можна прийняти рівним середній тривалості однієї телефонної роз-

мови, тобто приблизно 5 хв. Середній час проковзувань 1 год (12 п'ятихвилинних інтервалів).

Передача даних зі швидкістю 64 кбіт/с супроводжується значно меншою надлишковістю, ніж передавання мовлення; помилки виявляються й усуваються пристроєм захисту від помилок. Таким чином, до передавання даних зі швидкістю 60, 1200, 2400, 4800 Бод по цифрових каналах 64 кбіт/с не ставляться високі вимоги щодо усунення проковзувань.

Проковзування сприймаються як помилки в ЦСЕ й зумовлюють помилковий прийом блока даних, а отже, його повторення. Тоді очевидно, що норми щодо частої появи проковзувань повинні бути не нижчі, ніж норми частоти надходження помилок, які регламентовані міжнародними рекомендаціями і стандартами: менше ніж 10 % усіх однохвилинних інтервалів можуть містити більше ніж чотири помилки, тобто в більш ніж 90 % всіх однохвилинних інтервалів повинно з'являтися не більш ніж по чотири помилки в ЦСЕ. Однак через легкість керування появою проковзувань можна вимагати, щоб вони з'являлися із частістю, що дорівнює 1/10 частости надходження помилок у ЦСЕ. Така обставина послаблює зазначену норму для помилок ЦСЕ: припустимо мати приблизно чотири проковзування за 10 хв. Ця норма повинна бути зменшена для передавання даних у вигляді групових сигналів: проковзування повинні з'являтися з інтервалами не менш ніж 15 хв. Тут проковзування в груповому сигналі спричиняють втрату циклового синхронізму в приймачі групового сигналу, що у свою чергу зумовлює повторну синхронізацію або аварійний стан комутаційної станції.

Вплив, привнесений проковзуваннями у разі передавання тексту й факсимільних сигналів, принципово несуттєво відрізняється від такого впливу на передавання даних.

У разі передавання сигналів керування й сигналізації комутаційних станцій (код № 7) одне проковзування з малою імовірністю може неправильно сприйнятися й зумовити втрату сигналу, як наслідок — неправильне встановлення з'єднання або передчасне роз'єднання.

Проковзування, що викликають шум (клацання) при передаванні мовлення або помилки під час передавання даних, можна усувати зсувом імпульсів у паузи при передаванні мовлення або в інтервали між блоками при передаванні даних шляхом збільшення ємності буферної пам'яті.

Кожному типу переданої інформації відповідає допустиме число проковзувань. Самий чутливий до проковзування тип електрозв'язку визначає максимально допустиме число проковзувань.

Проковзування, що виникають у різному синхронному обладнанні, можна зменшити вибором більш якісних ТГ і буферної пам'яті. У міжнародних рекомендаціях і стандартах для зменшення до мінімуму числа проковзувань у цифрових закінченнях визнано доцільним використовувати два типи вирівнювального обладнання: вирівнювачі циклів і часових інтервалів. У разі використання вирівнювачів циклів проковзування додаються або виключаються символи, які ідуть один за одним і утворюють цикл. При використанні вирівнювачів часових інтервалів проковзування додаються або виключаються вісім символів, які ідуть один за одним, канального часового інтервалу в одному або декількох каналах 64 кбіт/с.

Таким чином, синхронізація мережі — це узагальнена концепція щодо засобу розподілення однакового для всіх вузлів часу або однакової частоти. Сучасні цифрові транспортні мережі потребують для розподілення опорних сигналів спеціальної мережі підтримки — мережі синхронізації. У загальному вигляді мережа синхронізації складається з вузлів, з'єднаних каналами синхронізації. Одна із задач цифрової мережі — гарантування отримання однієї і тієї самої швидкості передачі інформації в цифрових мережах. Усі генератори, що встановлені на цифровій мережі, повинні синхронізуватися від одного або кількох генераторів з близькими значеннями частот вихідних коливань.

21.4. ТРАДИЦІЙНІ МЕТОДИ ВИДІЛЕННЯ ТАКТОВИХ СИНХРОСИГНАЛІВ

Широкого розповсюдження набув спосіб демодуляції цифрових даних, що забезпечується імпульсним стробуванням цифрового сигналу електрозв'язку. Формування на приймальній стороні в демодуляторі стробувальних (опитувальних) імпульсів здійснюється на базі тактового синхросигналу (ТСС) системою тактової синхронізації (СТС).

Унаслідок просторової віддаленості передавача (кодера, модулятора) і приймача (декодера, демодулятора) виникає задача вирішення часових невизначеностей в місці прийому — формування ТСС. У більшості випадків ТСС в цифрових системах переміщення інформації (ЦСПІ) отримують за допомогою видільників тактових синхросигналів (ВТС), які виділяють ТСС із цифрового сигналу електрозв'язку, що прийшов із лінії. Технологія обробки випадкового лінійного ЦСЕ зводиться до виділення прихованої в ньому регулярної складової та формуванні ТСС із заданими якісними показниками.

Традиційні способи виділення ТСС із ЦСЕ в основному базувались на теорії фільтрації. Технічною реалізацією ВТС з цієї точки зору були резонансні контури, контури вибухових збуджень, вузькосмугових фільтрів (ВСФ) і т.п. Таким ВТС притаманні недоліки принципового характеру. Вони забезпечували роботу ЦСПІ при просторовому і частотному переміщенні інформації на низьких швидкостях. Збільшення швидкості переміщення інформації до декількох десятків, сотень, тисяч мегабіт за секунду порушувало якість отриманої інформації або зовсім зривало організацію зв'язку. Крім того, ситуація ще більше ускладнилась з упровадженням сучасних високошвидкісних ЦСПІ, які використовують недвійкові коди і відповідні їм багаторівневі сигнали.

Виділення тактового синхросигналу — це отримання тактового сигналу із сигналу, який приймається. Видільник ТСС — пристрій, який виділяє синхросигнал із сигналу, який приймається. Цей метод реалізується двома способами: 1) виділенням із інформаційного сигналу спеціально наявних у ньому синхросигналів (пілот-сигналів); 2) виявленням прихованої в інформаційному сигналі синхроінформації. Таким чином, один із головних недоліків цих двох способів виділення ТСС закладено в самій реалізації — в спектрі лінійного інформаційного сигналу повинні бути сигнали тактової частоти. Проте ця вимога не завжди виконується: енергетичні спектри лінійних цифрових сиг-

налів (бінарних, різні види трійкових, квазітрійкових і інших багаторівневих сигналів, сигналів з парціальним відкликом) не мають складової тактової частоти. У цьому випадку потрібні інші методи виділення ТСС із інформаційного.

21.4.1. Резонансний метод виділення тактового синхросигналу

Теоретичним підґрунтям резонансного методу є подання енергетичного спектра випадкового імпульсу некорельованого сигналу $S(t)$ (рис. 75) у вигляді неперервної $F_н(\omega)$ і дискретної $F_д(\omega)$ складових:

$$F(\omega) = \sigma^2 F_н(\omega) + a^2 F_д(\omega), \qquad (120)$$

де σ^2 — дисперсія випадкових амплітуд імпульсів; a — середнє значення імпульсної послідовності.

Тактові імпульси (синхросигнал) виділяються безпосередньо із послідовності інформаційних сигналів (символів) високовибірковими вузькосмуговими фільтрами (ВСФ), які в більшості випадків є резонансними контурами, що настроєні на тактову частоту $f_т$.

Відповідно до виразу (120) амплітуда гармонік дискретної складової $F_д(\omega)$ прямопропорційна квадрату середнього значення імпульсного сигналу (a^2). В спектрі групового ЦСЕ з імпульсно-кодовою модуляцією, який є послідовністю двополярних інформаційних сигналів із середнім значенням $a = 0$, відсутня дискретна складова з частотою $f_т$.

Як наслідок, із цього випливає необхідність з метою селекції синхроімпульсів домогтися, щоб середнє значення послідовності інформаційних сигналів було відмінним від 0 ($a \neq 0$). У цьому випадку виникає дискретна складова спектра $F_д(\omega)$, яка є набором гармонік, кратних основній частоті слідування імпульсів $\omega_т = 2\pi f_т$. Мета досягається нелінійним перетворенням двополярних сигналів в однополярні з $a \neq 0$.

Типова апаратурна реалізація резонансних пристроїв тактової синхронізації відповідно до викладеної вище ідеї виділення синхроімпульсів можлива за структурною схемою, яка відображена на рис. 76. Дві послідовності тактових імпульсів $ТІ_1$ і $ТІ_2$, які необхідні для пристроїв реєстрації окремих символів ЦСЕ, формуються шляхом проходження послідовності інформаційних сигналів через каскадне з'єднання диференціюючого елемента $ДЕ_1$, двопівперіодного випрямляча, підсилювача потужності, резонансного контуру, обмежувача, диференціюючого елемента $ДЕ_2$ та формуючого пристрою.

Суттєвість способу формування тактової частоти полягає в виділенні регулярної складової із інформаційного сигналу, енергетичний спектр якого

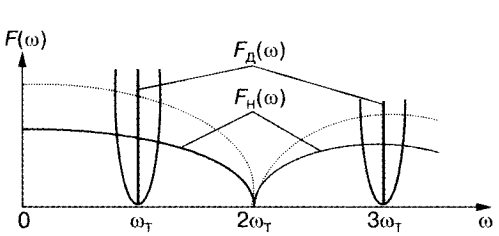


Рис. 75. Енергетичний спектр випадкового імпульсу сигналу $F(\omega)$

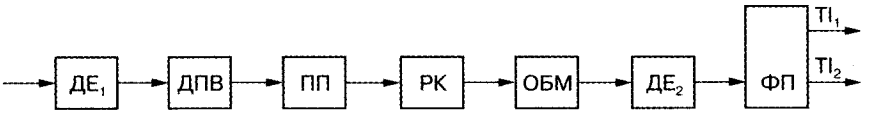


Рис. 76. Структурна схема резонансного пристрою виділення тактового синхросигналу: $ДЕ_1$ і $ДЕ_2$ — відповідно 1- і 2-й диференціюючі елементи: ДПВ — двопівперіодний випрямляч; ПП — підсилювач потужності; РК — резонансний контур; ОБМ — обмежувач; ФП — формуючий пристрій; $ТІ_1$ і $ТІ_2$ — відповідно перша та друга послідовність імпульсів

можна розкласти на неперервну і дискретну частини. Неперервній частині спектра відповідає випадкова складова, а дискретній — регулярна. Пристрій формування тактового синхросигналу включає в себе послідовно з'єднані двопівперіодний випрямляч, підсилювач, вузькосмуговий фільтр, пристрій формування коротких імпульсів з частотою слідування $f_т$. У цьому пристрої нелінійний перетворювач, який об'єднує випрямляч та підсилювач, виділяє дискретну складову спектра випадкової вхідної послідовності інформаційних імпульсів, а вузькосмуговий фільтр формує коливання основної частоти слідування імпульсів.

Резонансним методам виділення тактового синхросигналу притаманні недоліки принципового характеру, які закладені в самому способі виділення ТСС. У першу чергу до них належить можливість втрати синхронізації, зокрема, у разі коротких перерв зв'язку, "завмирань" у каналах зв'язку унаслідок зриву коливань на виході резонансного контуру; можливість збоїв в роботі апаратури в цілому за наявності довгих послідовностей символів без зміни їх фронтів у структурі вхідного ЦСЕ, з цієї причини забороняється використовувати кодові групи з тривалою відсутністю змін фронтів імпульсів або виникає потреба застосовувати операції скремблювання-дескремблювання. Крім того, таким ВТС притаманна низька завадостійкість через одержання на виході контуру "квазігармонічних", а не гармонічних коливань, оскільки на його вхід надходить випадкова послідовність збуджувальних імпульсів, а також мала точність фазування унаслідок паразитних фазових тремтінь, оскільки неперервна частина спектра вхідного ЦСЕ попадає в смугу пропускання фільтра і відіграє роль завади (див. рис. 75).

21.4.2. Виділення ТСС з цифрових багаторівневих сигналів

Цифрові сигнали до передачі в лінію кодуються так званим лінійним кодом. При двійковому кодуванні число рівнів вхідного сигналу — $m = 2$ і число рівнів вихідного сигналу — $n = 2$ (дворівневе кодування) або $n = 3$ (трирівневе кодування). У свою чергу дворівневе кодування є однополярним (+1, 0) і двополярним, симетричним (+1, –1); а трирівневе — однополярним (+2, +1, 0) і двополярним (+1, 0, –1). "ОДИНИЦЮ" можна подати позитивним імпульсом з повним заповненням тривалості інтервалу або типу "меандр"; чи переходом всередину інтервалу з "+1" на "0" або "–1". "НУЛЬ" подається відповідною тривалістю сигналу всередині інтервалу від "–1" до "+1".

Для обмеження тривалості послідовності тактових інтервалів без змін (послідовності "ОДИНИЦЬ" або "НУЛІВ") використовується інверсія — зміна полярності імпульсів регулярної кодової послідовності. Можна використовувати вставки додаткових двійкових символів визначеної полярності для збереження паритету кодової комбінації.

Широкого розповсюдження в ЦСПІ набули такі типи лінійних кодів.

• 1B2B — один біт вихідної послідовності тривалістю T кодується комбінацією із двох бітів тривалістю $T/2$, що призводить до збільшення у два рази швидкості переміщення інформації в каналі зв'язку. Прикладами цього коду можуть бути коди СМІ і Міллера (рис. 77, *ж*, *з*).

• ADI (Alternate Digital Inversion Code) — бінарний код з інверсією полярності сигналу на кожному бінарному розряді (незалежно "1" або "0" знаходиться у ньому), тобто це двополярний дворівневий код з інверсією, що чергується порозрядно (див. рис. 77, *д*).

• AMI (Alternate Mark Inversion Code) — бінарний код RZ з інверсією на кожній "1", тобто це двополярний код з поверненням в "0", унаслідок чого формується двополярний трирівневий код. Він може бути створений з коду ADI шляхом інверсії кожної парної "1" (див. рис. 77, *е*).

• B3ZS (Bipolar with 3 Zero Substitution Code) — біполярний код з підстановкою альтернативних блоків замість блоків з трьох "0", тобто замість блока "000" підставляється блок "00V" або "B0V" для збереження паритету — аналог коду HDB2 (V — інверсія, B — вставка).

• B6ZS (Bipolar with 6 Zero Substitution Code) — біполярний код з підстановкою альтернативних блоків замість блоків з шести "0", тобто замість блока "000000" підставляється блок "0VB0VB".

• B8ZS (Bipolar with 8 Zero Substitution Code) — біполярний код з підстановкою альтернативних блоків замість блоків з восьми "0", тобто замість блока "00000000" підставляється блок "000VB0VB";

• СМІ (Coded Mark Inversion Code) — дворівневий без повернення в "0" бінарний код класу 1B2B з інверсією полярності кодової комбінації на повний інтервал кожної "1", тобто кожній "1" відповідає комбінація "11" або "00" зі зміною полярності у середині кожного інтервалу "0", тобто кожному "0" відповідає "01" (див. рис. 77, *ж*).

• HDB2 (High Density Code of order 2) — двополярний код високої щільності порядку 2 — код RZ з інверсією на "1" (аналогічно АМІ), у якому кожен блок "000" заміняється на блок "00V" або "B0V", при цьому вставка B імпульсу "1" здійснюється так, щоб кількість B імпульсів між послідовними V імпульсами була непарною. Вказана процедура формує трирівневий код.

• HDB3 (High Density Code of order 3) — двополярний код високої щільності порядку 3 — код RZ з інверсією на "1", у якому кожен блок "0000" заміняється на блок "000V" або "B00V", при цьому вставка B імпульсу "1" здійснюється так, щоб кількість B імпульсів між послідовними V імпульсами була непарною. Вказана процедура формує трирівневий код (див. рис. 77, *і*).

• NRZ (Non Return to Zero Code) — двополярний або однополярний дворівневий код без повернення до "0" (див. рис. 77, *б* і *в*).

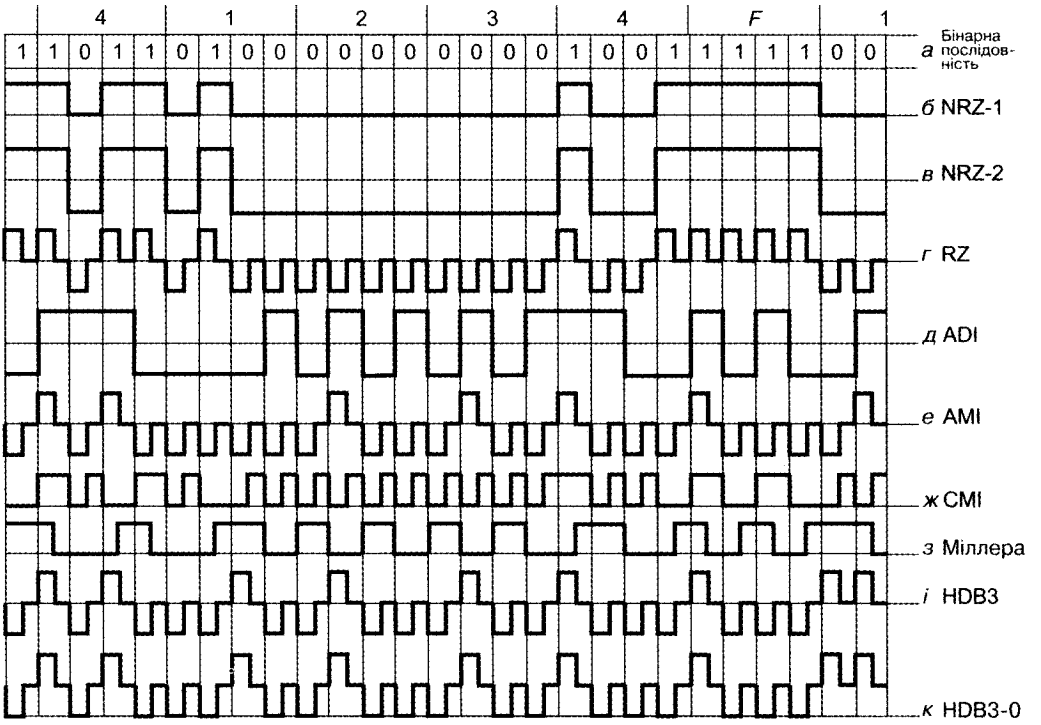


Рис. 77. Типи лінійних кодів

• RZ (Return to Zero Code) — двополярний трирівневий код з поверненням до "0" (див. рис. 77, *г*).

• mBnB — загальна формула блочних кодів (m — довжина у бітах блоків, на які розбивається початкова цифрова послідовність; n — довжина, що їм відповідає, у бітах блоків, складених із кодових символів, B — біт).

• Miller Code — двополярний дворівневий код Міллера класу 1B2B, який має багато станів {00, 01, 10, 11}.

У ЦСПІ залежно від співвідношення швидкості передачі при переміщенні цифрових потоків і смуги пропускання каналів зв'язку використовують бінарні, різні типи трійкових, квазітрійкових та інших багаторівневих сигналів, а також сигнали з парціальним відкликом. Енергетичні спектри таких сигналів дуже часто не містять складової тактової частоти. Наприклад, трійковий код 6B4T дає змогу збільшити питому швидкість передачі інформації в 1,5 раза. Енергетичний спектр сигналу коду 6B4T має максимум при $F_{max} = 0{,}2\ f_т$ ($f_т$ — тактова частота). Сигнал коду 6B4T містить постійну складову і не містить в явному вигляді дискретної складової $f_т$; тривалість входження в блочний синхронізм — $t_{бл} = 0{,}5$ с з імовірністю, що дорівнює 0,96. Сигнал на вході пристрою перетворення коду є послідовністю прямокутних імпульсів з тривалістю, яка дорівнює тривалості тактового інтервалу. Висока питома швидкість передачі інформації на одиницю смуги частот каналу зв'язку призводить до по-

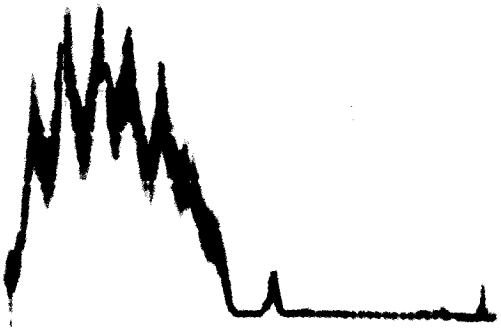


Рис. 78. Спектрограма п'ятирівневого сигналу

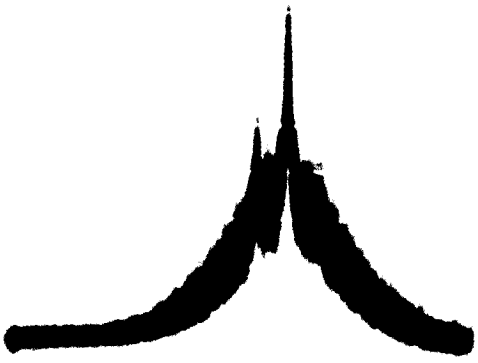


Рис. 79. Спектрограма сигналу на виході ВТС

шуку допоміжної обробки багаторівневого сигналу, який надходить з лінії, та до збільшення числа його рівнів. Як бачимо зі спектрограми (рис. 78), енергетичний спектр такого п'ятирівневого сигналу не містить складової тактової частоти.

Таким чином, ціль видільника тактового синхросигналу — обробка багаторівневого сигналу, який приходить з лінії, з метою виділення тактового синхросигналу. Як приклад на рис. 79 наведено спектрограму сигналу на виході ВТС.

21.5. ЗАВАДОСТІЙКЕ ВИДІЛЕННЯ ТАКТОВИХ СИНХРОСИГНАЛІВ

21.5.1. Постановка задачі виділення тактових синхросигналів із ЦСЕ на базі теорії прийняття оптимальних рішень

Традиційні способи виділення ТСС із ЦСЕ головним чином базувалися на теорії фільтрації. Необхідність застосування в сучасних високошвидкісних ЦСПІ недвійкових кодів і відповідних їм багаторівневих сигналів, які дають можливість більшою мірою використовувати середовище поширення, не завжди дає змогу виконувати якісне виділення ТСС. У свою чергу ця ситуація, з одного боку, підвищила вимоги щодо часового положення моментів рішення, що задаються СТС, стосовно значущих моментів ЦСЕ, а з іншого — призвела до збільшення фазових тремтінь цифрового сигналу на виході СТС. Фазові тремтіння залежать перш за все від статистик ЦСЕ, що приймається, його коду, числа рівнів, щільності заповнення інформаційним імпульсом тактового інтервалу та інших факторів.

Задача видільника тактового синхросигналу — обробка цифрового сигналу електрозв'язку, який надходить з лінії, з метою завадостійкого виділення тактових синхросигналів.

Для математичної постановки задачі оптимального виділення тактових синхросигналів ЦСЕ подамо аналітично випадкову однополярну послідов-

ність імпульсів на вході ВТС у вигляді

$$И(t) = \sum_{i=0}^{N} \alpha_i a_i (t_i + \tau_i), \qquad (121)$$

де $t_i = t_i' \pm \Delta t_i$ — час надходження i-го імпульсу; $t_i' = iT, i = \overline{1, N}$; T_i — період i-го тактового інтервалу; для синхронних систем зв'язку $T_i = T = \text{const}$; Δt_i — часовий зсув i-го імпульсу відносно i-го значущого моменту t_i', $0 \le \Delta t_{ii} \le \frac{T}{2}$; a_i — випадкова величина, яка приймає з вірогідністю p_{ai} значення, що дорівнює "1", якщо є i-й імпульс, і "0" — з вірогідністю $(1 - p_{ai})$ у випадку відсутності i-го імпульсу (a_i відображає структуру кодового рисунка); $a_i = a_n \pm \Delta a_i$ — амплітуда i-го імпульсу; a_n — середнє значення амплітуди; Δa_i — випадкове значення флуктуацій амплітуди; $\tau_i = \tau_И \pm \Delta\tau_i$ — тривалість i-го імпульса; $\tau_И$ — середня тривалість i-го імпульсу $0 \le \Delta\tau_i \le T$; $\Delta\tau_i$ — випадкове подовження чи скорочення i-го імпульсу $0 \le \Delta\tau_i \le T$; N — випадкове число імпульсів.

У подальшому без спеціальних застережень будемо вважати, що початок відліку (поява першого імпульсу) збігається з початком тактового інтервалу T.

Параметри послідовності імпульсів (121) Δt_i, α_i, Δa_i, $\Delta\tau_i$, N — є випадковими величинами, що дає право називати її випадковим процесом, який являє собою сукупність (ансамбль) випадкових функцій часу. Таким чином, за математичну модель сигналу на вході ВТС будемо вважати імпульсний процес з детермінованим тактовим інтервалом, який в загальному випадку не є стаціонарним.

Задача виділення ТСС із ЦСЕ. Нехай задано випадковий процес $\{И(t)\}$ відповідно до виразу (121), який є наперед невідомим поєднанням інформаційного ЦСЕ і шуму; а його реалізації $И(t)$ повністю визначені на просторі. Нехай також α_i — випадкове значення i-го імпульсу, який з імовірністю p_a становить $d = 1$, якщо є i-й імпульс, і $d = 0$, з імовірністю $(1 - p_a)$, якщо він відсутній, що виступає як параметр, що належить відомій області Ω_n. Допустимо, що спостерігається скінченна сукупність випадкових величин $\{u(t_i)\}$, $i = 1, N$, яка повністю описується N-вимірною функцією розподілу імовірностей $F(u/\alpha_i)$ залежно від α_i.

Існує вимога: за тривалість тактового інтервалу $T_И$ за результатами спостереження реалізації $u(t)$ згідно з правилом розв'язування $П \in П_д$ із класу допустимих, яке відображає множину Г в множину D розв'язків $П : Г \Rightarrow D$, прийняти оптимальний розв'язок $d = П(u)$, $d \in D$, для кожної можливої реалізації $U \in Г$, для забезпечення мінімуму функції ризику $r(\alpha_i, П) = M[Ц[\alpha_i, П(U/\alpha_i)]]$, де $M[Ц[\alpha_i, П(u/\alpha_i)]]$ — математичне очікування випадкової функції втрат $Ц(\alpha_i, d) = Ц[\alpha_i, П(u)]$, поданої у вигляді матриці втрат:

$$Ц(\alpha_i, d) = \begin{bmatrix} Ц(0, d_0) & Ц(0, d_1) \\ Ц(1, d_0) & Ц(1, d_1) \end{bmatrix}. \qquad (122)$$

Вимагається найбільш "чисто" виявити переданий сигнал $И(t)$ або $c_0(t, \lambda_0)\backslash c_1(t, \lambda_1)$ на фоні випадкового шуму і завади та прийняти рішення, який із символів цифрової послідовності на вході ВТС у даній реалізації має місце ("1" чи "0"), після цього відповідно до функції регулярності виділити регулярну складову ЦСЕ і за її допомогою сформувати ТСС.

21.5.2. Алгоритм завадостійкого виділення ТСС

Розглянемо алгоритм завадостійкого виділення ТСС із повністю відомого ЦСЕ.

Нехай про ЦСЕ, який надходять з лінії на вхід ВТС, відомо все. На підставі аналізу сигналу $u(t)$, який прийнято згідно з функцією рішення $П(u)$, приймаємо рішення: $П(u) = 1$, якщо сигнал є або $П(u) = 0$, що засвідчує наявність альтернативного рішення. Остання обставина потребує генерації сигналу $c_1 = 1$.

Таким чином, структура оптимального ВТС за цілком відомого сигналу, що надходить із лінії, повинна бути такою, щоб можна було розрахувати відношення правдоподібності (ВП) $Л'(u)$ та порівняти його з пороговим значенням $Л'_0$.

Розрахунок відношення правдоподібності — складна задача, яка залежить від вигляду періодичної послідовності, що генерується головним тактовим генератором, спотворень, яких зазнають імпульси даної послідовності в каналі зв'язку під впливом шумів і завад та ін. Вигляд періодичної послідовності, що генерується головним тактовим генератором, спотворення імпульсів вказаної послідовності в каналі зв'язку під впливом шумів та завад і т.п. визначають складність розрахунку відношення правдоподібності.

Відомо, що клас правил рішень, що базуються на порівнянні ВП з порогом, є повним і однотиповим. Різниця між правилами рішень та правилами прийняття рішень залежить від вибору критерію оптимальності (Бейєса, Неймана—Пірсона, мінімаксного та ін.) і впливає лише на значення порога $Л'_0$.

Згідно з критеріями оптимальності, які зводяться до правила рішення з розрахунком ВП, видільник тактового синхросигналу повинен формувати ВП $Л'(u)$ і зрівнювати його з пороговим значенням $Л'_0$ з метою прийняття одного із альтернативних рішень: корисний сигнал відсутній або корисний сигнал на вході ВТС є. Формувач порога (ФП) формує поріг $Л'_0$, значення якого залежить від роботи ВТС і вибраного критерію оптимальності (Бейєса, Неймана—Пірсона, мінімаксного та інших). Структура формувача відношення правдоподібності головним чином залежить від вигляду та статистичних характеристик інформаційного ЦСЕ.

Якщо умовно вважати інформаційні параметри ТСС детермінованими, то в цьому випадку можна запропонувати варіант виділення періодичного ТСС із ЦСЕ за допомогою ВТС, реалізованого на базі визначення безумовного відношення правдоподібності (БВП). Як альтернативний розв'язок пропонується при виділенні ТСС із випадкового ЦСЕ інформаційні параметри ТСС вважати випадковими і здійснювати статистичне усереднення виділеного ТСС шляхом вторинної обробки ТСС, наприклад, за допомогою системи фазової автоматичної настройки частоти (ФАНЧ).

21.5.3. Функціональна схема завадостійкого виділення ТСС

Удосконалення методів виділення ТСС із інформаційного ЦСЕ та розробка нових зумовлені такими чинниками:

1) складністю або неможливістю досягнення за допомогою традиційних методів потрібної якості роботи СТС за високих швидкостей переміщення інформації;

2) розширеними можливостями сучасної елементної бази побудови СТС.

Якщо перша причина потребує перегляду статистик вихідних даних, на яких виконується аналіз, та врахування впливу зсуву фази ТСС із-за значних величин фазових тремтінь і дії дестабілізувальних факторів, то друга дає змогу використати сучасний комплексний підхід до оптимізації не тільки окремих вузлів СТС, а й всієї її структури, а також забезпечити оптимальну обробку ЦСЕ та реалізувати СТС з єдиним оптимальним алгоритмом роботи на сучасній елементній базі (включно) та з застосуванням великих інтегральних схем і мікропроцесорів.

Під час використання методу виділення ТСС із ЦСЕ статистики кодового рисунка ЦСЕ та вплив дестабілізувальних факторів (зміна значень тактової частоти, напруги джерел живлення, температури навколишнього середовища і т.д.) враховуються за допомогою процедури оптимальної обробки ЦСЕ на базі статистичної теорії прийняття рішень і методів оптимального керування.

Розглянемо оптимальний алгоритм функціонування СТС на базі завадостійкого виділення ТСС. Він грунтується на оптимальній обробці ЦСЕ на базі статистичної теорії прийняття рішень, унаслідок якої виділяється регулярна складова ЦСЕ (первинна обробка сигналу). Технічну реалізацію цієї процедури названо видільник регулярної складової (ВРС).

Таким чином, ВТС, який є частиною СТС, відповідно до реалізації оптимального алгоритму повинен включати в себе: 1) вхідні пристрої (ВП) для підключення ВТС до приймача; 2) ВРС для виділення регулярної складової, що прихована в випадковому ЦСЕ; 3) систему ФАНЧ для порівняння сигналів з виходу ВРС і веденого тактового генератора (ВдТГ) та автоматичної настройки його фази сигналу під фазу сигналу з виходу ВРС з допомогою системи ФАНЧ. Технічна реалізація такого оптимального ВТС наведена на рис. 80.

Випадковий ЦСЕ формується в загальному каналі (ЗК ЦСЕ) під дією ведучого тактового генератора (ВТГ) і передається в лінію зв'язку. Вхідний пристрій приймача ЦСПІ (ВхПРГ) виконує попередню обробку ЦСЕ, що надійшов із лінії, і згідно з будь-яким статистичним критерієм (Бейєса, Неймана—Пірсона, мінімаксним та ін.) формує сигнал відношення правдоподібності $Л(u)$ у блоці ФВП. Сигнал відношення правдоподібності подається на пороговий пристрій (ПП), на виході якого приймається рішення про наявність символу у випадковій імпульсній послідовності ЦСЕ або про його відсутність. У подальшому після інверсії в блоці І та об'єднання в блоці логічного додавання (ЛД) виділяється регулярна складова ЦСЕ. Блоки І та ЛД утворюють ВРС. У фазовому детекторі, що управляється (УФД) порівнюються сигнали з виходу ВРС і веденого (Вд ТГ). Якщо виявляється розходження їх фаз,

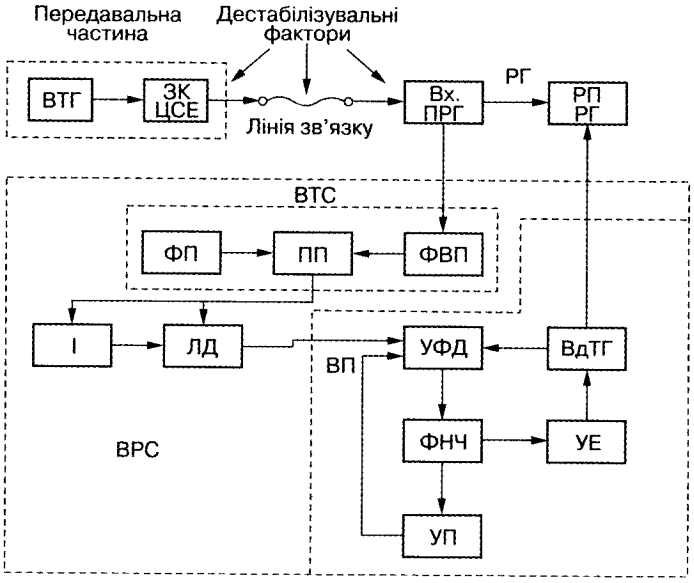


Рис. 80. Функціональна схема завадостійкого виділення ТСС

то формується сигнал, пропорційний цьому розходженню, який проходить крізь фільтр нижніх частот (ФНЧ) та елемент управління (УЕ) і підлаштовує фазу сигналу ВдТГ під фазу сигналу, який знімається з виходу ВРС. Пристрій управління (УП), який є логічним обчислювальним блоком, на базі обробки контрольованих значень фазової помилки та її похідних виробляє сигнал управління характеристикою УФД, чим і забезпечується протікання оптимальних за швидкодією режимів перехідних процесів. Виділений ТСС надходить на пристрій приймача, що приймає рішення (РП РГ).

ВИСНОВКИ

Цю працю присвячено одному з найважливіших розділів теорії інформації — кодам, які контролюють помилки.

Теорія кодування — це прикладна наука. Вона вирішує питання, пов'язані з технікою зв'язку, радіолокацією, вимірювально-обчислювальною технікою. Тому велике значення мають застосовність отримуваних результатів та їх переваги над некодовими методами захисту від помилок. Нині добре відомі практичні досягнення теорії кодування, проте найбільш застосовними є найпримітивніші способи кодування, такі як перевірка на парність, коди Хеммінга, циклічні коди, прості згортальні коди і тому подібне

З погляду фахівців з теорії кодування, відставання у використанні більш потужних і нетривіальних кодів пов'язане зі складністю сприйняття інженерами математичного апарату теорії й із недостатньою розробкою питань взаємозв'язку кодування і модуляції.

Автори сподіваються, що ця праця певною мірою сприятиме подоланню такого відставання, тому і математичні питання викладено досить повно з наведенням прикладів і завдань.

Теорія алгебри кодування містить багато результатів, які заслуговують докладного розгляду. Основну увагу в нашій праці акцентовано на поясненні різних методів теорії кодування, відповідно до чого й подано найбільш детально досліджені коди БЧХ і методи їх декодування, важливі з практичної точки зору, поліноміальні коди, що містять як свої підкоди коди БЧХ, а також деякі інші класи кодів. Крім того, наведено інтенсивно досліджувані в даний час коди Ріда—Соломона, що виправляють пакети помилок.

Автори не претендують на вичерпний виклад усього матеріалу теорії кодування, їх мета — розглянути всі найбільш важливі питання в розвитку методів теорії.

Під час попереднього ознайомлення з теорією кодування труднощі, як правило, виникають там, де використовуються поняття і результати теорії скінченних полів. Тому до праці включено розділ, в якому стисло розглядаються основні поняття і результати теорії скінченних полів, використовуваних в теорії кодування. Для розуміння змісту праці достатньо володіння університетським курсом математики. Багато результатів теорії кодування формулюються у вигляді завдань або тверджень.

Предмет кодування, яке контролює помилки, одночасно простий і складний. Він простий в тому сенсі, що його завдання легко пояснити будь-якій технічно грамотній людині, а складний — оскільки, для того щоб отримати розв'язок задачі (навіть частковий), потрібно осягнути увесь матеріал цього підручника, а також мати знання з деяких розділів сучасної алгебри.

У праці першочергове значення надається реалізації кодерів і декодерів за допомогою ланцюгів, що містять регістри зсуву, і переважно використовується термінологія, прийнята в галузі цифрової обробки даних. Для найкращого розуміння викладу авторами описано побудову регістрів зсуву і виявлено модифікації, що дають змогу зменшити кількість елементів у пристроях. Навіть у випадках, коли описувалася програмна реалізація пристроїв, наводилися деякі міркування стосовно того, як, використовуючи регістр зсуву, можна спростити програму. Автори повністю погоджуються з Р. Блейхутом, а саме: остаточним критерієм якості коду або алгоритму роботи пристрою є вартість кодера і декодера. Розробник пристрою не цікавитиметься кодами з найбільшою мінімальною відстанню, якщо невідомі якісні алгоритми їх декодування. Пошук нових кодів, що допускають використання відомих алгоритмів декодування, часто виявляється плідніщим, ніж розробка нових алгоритмів декодування для відомих кодів.

СПИСОК ЛІТЕРАТУРИ

1. *Абдуллаев Д.А., Арипов М.Н.* Передача дискретных сообщений в задачах и упражнениях: Учеб. пособие для вузов. — М.: Радио и связь, 1985. — 128 с.

2. *Айфичер Э.С., Джервис Б.У.* Цифровая обработка сигналов: практический подход: Пер. с англ. — 2-е изд. — М.: Издательский дом "Вильямс", 2004.— 992 с.

3. *Баева Н.Н.* Основы многоканальной электросвязи. — М.: Связь, 1975. — 328 с.

4. *Блейхут Р.* Быстрые алгоритмы цифровой обработки сигналов. — М.: Мир, 1989. — 448 с.

5. *Блейхут Р.* Теория и практика кодов, контролирующих ошибки: Пер. с англ. — М.: Мир, 1986. — 576 с.

6. *Берлекэмп Э.* Алгебраическая теория кодирования. — М.: Мир, 1971. — 478 с.

7. *Винберг Э.Б.* Курс алгебры. — М.: Факториал Пресс, 2002. — 544 с.

8. *Габидулин Э.М., Афанасьев В.Б.* Кодирование в радиоэлектронике. — М.: Радио и связь, 1986. — 176 с.

9. *Галлагер Р.* Теория информации и надежная связь. — М.: Сов. радио, 1974. — 720 с.

10. *Гинзбург В.В., Каяцкас А.А.* Теория синхронизации демодуляторов. — М.: Связь, 1974. — 216 с.

11. *Глинченко А.С.* Цифровая обработка сигналов: В 2 ч. — Красноярск: Изд-во КГТУ, 2001. — 383 с.

12. *Гольденберг Л. М.* Цифровая обработка сигналов: Справочник. — М.: Радио и связь, 1985. — 312 с.

13. *Даджион Д., Мерсеро Р.* Цифровая обработка многомерных сигналов. — М.: Мир, 1988. — 488 с.

14. *Дадаев Ю.Г.* Теория арифметических кодов. — М.: Радио и связь, 1981. — 272 с.

15. *Зарисский О., Самюэль П.* Коммутативная алгебра: Пер. с англ.: В 2 т. — М.: Радио и связь, 1987. — Т. 1. — 287 с.; Т. 2. — 328 с.

16. *Злотник Б.М.* Помехоустойчивые коды в системах связи. — М.: Радио и связь, 1989. — 232 с.

17. *Касами Т., Токура Н., Ивадари Е., Инагаки Я.* Теория кодирования: Пер. с япон. — М.: Мир, 1978. — 576 с.

18. *Кларк-мл. Дж., Кейн Дж.* Кодирование с исправлением ошибок в системах цифровой связи: Пер. с англ. — М.: Радио и связь, 1987. — 392 с.

19. *Коржик В.И., Финк Л.М.* Помехоустойчивое кодирование дискретных сообщений в каналах со случайной структурой. — М.: Связь, 1975. — 272 с.

20. *Куприянов М.С., Матюшкин Б.Д.* Цифровая обработка сигналов: процессоры, алгоритмы, средства проектирования. — 2-е изд., перераб. и доп. — СПб.: Политехника, 1999. — 592 с.

21. *Мак-Вильямс Ф.Дж., Слоэн Н.Дж.А.* Теория кодов, исправляющих ошибки. — М.: Связь, 1979. — 331 с.

22. *Макс Ж.* Методы и техника обработки сигналов при физических измерениях: В 2 т. — М.: Мир, 1983. — Т. 1. — 234 с.; Т. 2. — 317 с.

23. *Марпл-мл. С.Л.* Цифровой спектральный анализ и его приложения. — М.: Мир, 1990. — 584 с.

24. *Морелос-Сарагоса Р.* Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение. — М.: Техносфера, 2005. — 483 с.

25. *Муттер В.М.* Основы помехоустойчивой телепередачи информации. — Л.: Энергоатомиздат. Ленингр. отделение, 1990. — 288 с.

26. *Немировский М.С.* Цифровая передача информации в радиосвязи // Статистическая теория связи. — М.: Связь, 1980. — Вып. 14 — 256 с.

27. *Оппенгейм А., Шафер Р.* Цифровая обработка сигналов. — 2-е изд., испр. — М.: Техносфера, 2007. — 856 с.

28. *Оппенгейм А. В., Шафер Р.В.* Цифровая обработка сигналов. — М.: Связь, 1979. — 416 с.

29. *Питерсон У.* Коды, исправляющие ошибки. — М.: Мир, 1964. — 278 с.

30. *Питерсон У., Уэлдон Э.* Коды, исправляющие ошибки. — М.: Мир, 1976. — 596 с.

31. *Прокис Дж.* Цифровая связь / Пер. с англ.; Под ред. Д.Д. Кловского. — М.: Радио и связь. 2000. — 800 с.

32. *Рабинер Л., Гоулд Б.* Теория и применение цифровой обработки сигналов. — М.: Мир, 1978. — 848 с.

33. *Самойленко С.И.* Помехоустойчивое кодирование. — М.: Наука, 1966. — 240 с.

34. *Скляр Бернард.* Цифровая связь. Теоретические основы и практическое применение: Пер. с англ. — 2-е изд., испр. — М.: Издательский дом "Вильямс", 2004. — 1104 с.

35. *Тактовая* синхронизация в интегральных цифровых сетях электросвязи / В.И. Борщ, В.П. Гайдар, В.В. Коваль, И.П. Лесовой. — Киев: Наук. думка, 1998. — 202 с.

36. *Теоретичні* основи завадостійкого кодування / П.Ф. Олексенко, В.В. Коваль, Г.М. Розорінов, Г.О. Сукач; [за ред. акад. НАН України В.Ф. Мачуліна]. — Київ: Наук. думка, 2010. — Ч. 1. — 192 с.

37. *Теория* прикладного кодирования: Учеб. пособие: В 2 т. / В.К. Конопелько, В.А. Липницкий, В.Д. Дворников и др.; Под ред. В.К. Конопелько. — Минск: БГУИР, 2004. — Т. 1. — 285 с.; Т. 2. — 398 с.

38. *Тихонов В.И.* Статистическая радиотехника. — М.: Сов. радио, 1982. — 624 с.

39. *Тихонов В.И.* Оптимальный прием сигналов. — М.: Радио и связь, 1983. — 220 с.

40. *Форни Д.* Каскадные коды. — М.: Мир, 1970. — 207 с.

41. *Хемминг Р.В.* Цифровые фильтры. — М.: Недра, 1987. — 221 с.

42. *Цифрова* обробка аудіо- та відеоінформації у мультимедійних системах: Навчальний посібник / О.В. Дробик, В.В. Кідалов, В.В. Коваль та ін. — Київ: Наук. думка, 2008. — 144 с.

ЗМІСТ

Навчальне видання

ОЛЕКСЕНКО Павло Феофанович
КОВАЛЬ Валерій Вікторович
РОЗОРІНОВ Георгій Миколайович
СУКАЧ Георгій Олексійович

ТЕОРЕТИЧНІ ОСНОВИ ЗАВАДОСТІЙКОГО КОДУВАННЯ

ЧАСТИНА II

Підручник для вищих навчальних закладів

Київ, Науково-виробниче підприємство
«Видавництво "Наукова думка" НАН України», 2012

Художній редактор *І.Р. Сільман*
Оформлення художника *В.В. Кузьменка*
Технічний редактор *Г.М. Ковальова*
Коректор *Ю.М. Кирпич*
Комп'ютерна верстка *Л.В. Багненко*

Підп. до друку 03.07.2012. Формат 70 × 100/16.
Папір офс. № 1. Гарн. Таймс. Друк. офс.
Ум. друк. арк. 17,23. Ум. фарбо-відб. 17,88.
Обл.-вид. арк. 15,0. Тираж 300 прим. Зам. 12-210

НВП «Видавництво "Наукова думка" НАН України»
Свідоцтво про внесення суб'єкта видавничої справи
до Державного реєстру ДК № 2440 від 15.03.2006 р.
01601 Київ 1, вул. Терещенківська, 3

ПП «Видавництво "Фенікс"»
03680 Київ 680, вул. Шутова, 13б