DonSTU
Computer
Department

КОНСПЕКТ ЛЕКЦІЙ З КУРСУ "ТЕОРІЯ КОРИГУЮЧИХ КОДІВ"

DonSTU
Computer
Department

Зміст

ТЕОРІЯ КОРИГУЮЧИХ КОДІВ

- Декодуючий пристрій коду Івадарі
- Векторні коди
- Графічне представлення кодів Івадарі

ТЕОРІЯ ІНФОРМАЦІЇ

- **ІНФОРМАЦІЙНІ МОДЕЛІ СИГНАЛІВ**
  - Кількісна міра інформації
  - Властивості безумовної ентропії дискретних повідомлень
  - Ентропія складних повідомлень. Ентропія об’єднання, умовна ентропія
  - Основні властивості ентропії складних повідомлень (ентропії об’єднання)

TEL. (062) 301-07-58
301-08-04
FAX. (062) 335-45-89
mailto:do@cs.dgtu.donetsk.ua

83000 г. Донецьк
вул. Артема 58
корпус 4, ауд. 4.14
кафедра “Комп’ютерна інженерія”

УДК 621.391

Конспект лекцій з курсу “Теорія коректуючих кодів” (для студентів спеціальності 7.091502 “Системне програмування”)/ Скл.: О.М.Дяченко - Донецьк: ДонНТУ, 2011. - 89 с. (на електронному носії № 193, прот. № 2 від 21.03.11)

Викладено основи теорії перешкодостійкого кодування інформації в системах зв'язку. Розглянуто найбільше широко розповсюджені в цей час методи кодування - з використанням блокових лінійних кодів, а також методи їхнього декодування.

Наведено велику кількість практичних завдань, що дозволяють закріпити досліджуваний матеріал.

В результаті вивчення дисципліни студенти отримають знання про тенденції розвитку науки та техніки в галузі перешкодостійкого кодування інформації; актуальні проблеми теорії перешкодостійкого кодування; основні терміни та визначення; принципи побудови кодів, що виявляють та виправляють помилки; способи побудови функціональних та принципових схем кодуючих та декодуючих пристроїв циклічних кодів; методи дослідження кодерів та декодерів, проведення порівняльного аналізу їх коригувальних здібностей; методи пошуку оптимальних рішень; математичні методи розв'язання задач, в тому числі і формалізованих методів, орієнтованих на використання ЕОМ.

Укладач: О.М.Дяченко

Рецензент: Ю.Є.Зінченко

ТЕОРІЯ КОРИГУЮЧИХ КОДІВ

Основні поняття і визначення. Задачі теорії інформації і кодування

Будь-яке відображення матеріального світу, що може бути зафіксовано живою істотою або приладом, несе в собі інформацію. Відображення результатів людської діяльності або розуміння навколишнього світу може бути представлене у формалізованому виді, наприклад, у виді наборів букв або цифр. Такі формалізовані набори звичайно називають даними.

Дані, отримані від джерела інформації, називають повідомленням. Інформацією стають ті повідомлення, що знімають невизначеність, що існувала до їхнього надходження.

Теорія інформації займається вивченням кількості інформації в повідомленнях безвідносно конкретного їхнього змісту, тому що процес формалізації і механізації передачі інформації не передбачає зміни змісту повідомлень. Предметом вивчення теорії інформації є імовірнісні характеристики досліджуваних об'єктів і явищ, тому що імовірність є найбільш зручна чисельна міра невизначеності, зі зменшенням якої і зв'язаний процес одержання інформації.

Інформація - це відомості, що є об'єктом збереження, передачі і перетворення.

Теорія передачі інформації є частиною теорії інформації. Предметом вивчення теорії передачі інформації є одержання оптимальних методів передачі повідомлень.

Повідомлення передаються за допомогою сигналів, що володіють визначеними фізичними властивостями. Розрізняють сигнали: зорові, акустичні, електричні, радіосигнали і т.д. Одні сигнали можуть викликати інші.

Основною задачею теорії інформації і кодування як самостійної дисципліни є оптимальне використання інформаційних характеристик джерел повідомлень і каналів зв'язку для побудови кодів, що забезпечують задану вірогідність переданої інформації з максимально можливою швидкістю і мінімально можливою вартістю передачі повідомлень.

Перешкодостійке кодування

Модель системи передачі інформації

Повідомлення передаються від об'єкта до адресата (від джерела до одержувача) за допомогою сукупності технічних засобів, що утворюють систему передачі інформації.

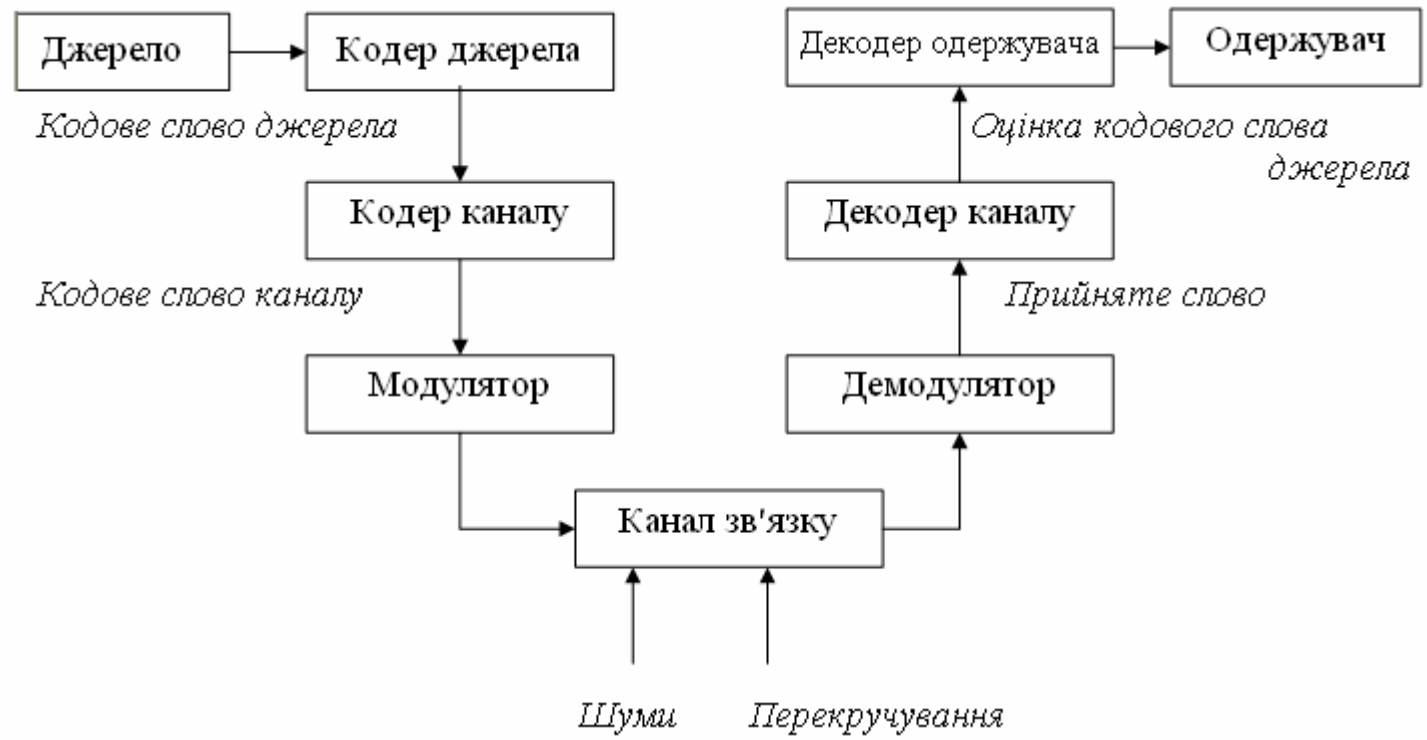
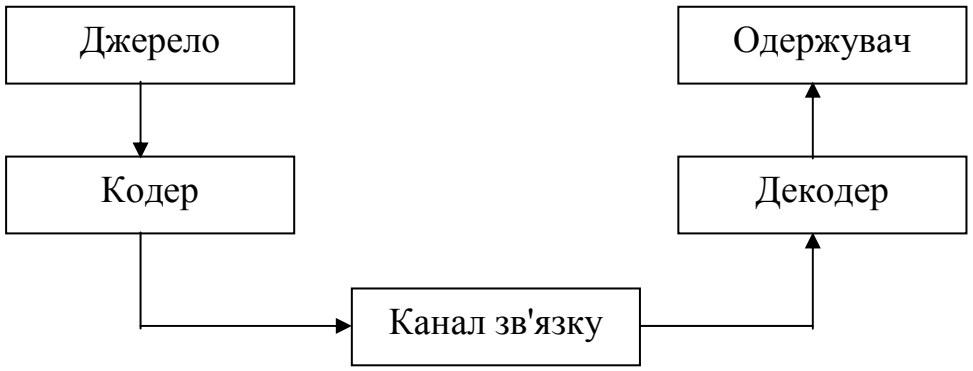


Схема типової цифрової системи зв'язку

Дані, що надходять у систему від джерела даних, насамперед, обробляються кодером джерела, призначеним для кодування вхідної інформації у двійкові символи. Це проміжне представлення даних джерела називається *кодовим словом джерела*. Далі дані обробляються кодером каналу, що перетворить послідовність двійкових символів кодового слова джерела в іншу послідовність двійкових символів, називану *кодовим словом каналу*. Кодове слово каналу являє собою нову, більш довгу послідовність з більшою, ніж у кодового слова надмірністю. Потім модулятор перетворить кожен символ кодового слова каналу у відповідний аналоговий символ із кінцевої безлічі припустимих аналогових символів: двійкові символи перетворяться в сигнали. Послідовність сигналів передається по каналу. Через те що в каналі виникають різного типу шуми, перекручування, вихідні сигнали каналу відрізняються від його вхідних сигналів. Демодулятор перетворює послідовність отриманих сигналів у послідовність символів одного з кодових слів каналу. Через шум у каналі демодулятор робить помилки. Демодульована послідовність символів називається *прийнятим словом*. Через помилки символи прийнятого слова не завжди відповідають символам кодового слова каналу. Декодер каналу використовує надмірність ко-

дового слова каналу для того, щоб виправити помилки в прийнятому слові, і потім видає оцінку кодового слова джерела. Якщо всі помилки виправлені, то *оцінка кодового слова джерела* збігається з вихідним кодовим словом джерела. Декодер джерела виконує операцію, зворотну операції кодера джерела, результат якої надходить до одержувача.

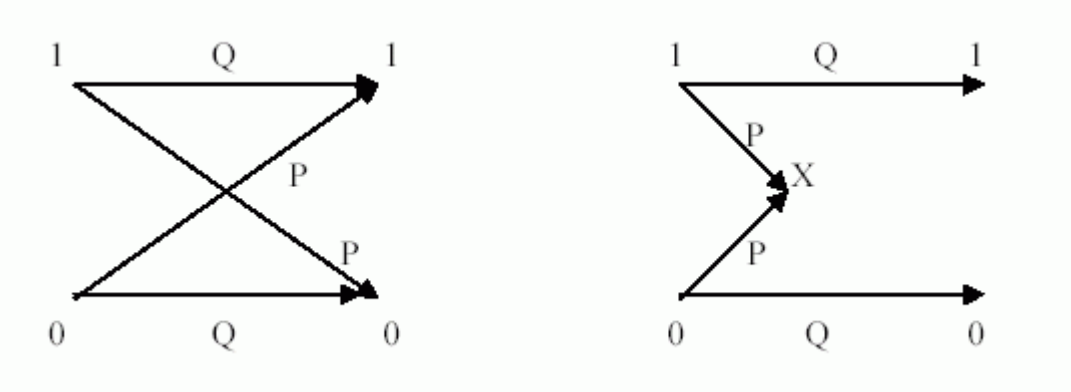
У теорії перешкодостійкого кодування переходять до спрощеної схеми, що містить кодер і декодер каналу, що називають просто кодером і декодером.



Спрощена схема типової цифрової системи зв'язку

Канали зв'язку

1. *Двійковий симетричний канал* *2. Двійковий канал, що стирає*



Для *двійкового симетричного каналу* задається імовірність Q того, що отриманий символ збігається з переданим. P = 1 - Q – імовірність одержання протилежного символу. Канал містить у собі модулятор, власне канал і може включати демодулятор.

Для *двійкового каналу, що стирає*, задається імовірність Q того, що буде отриманий той же символ, що передавався, і імовірність P == 1 - Q того, що переданий символ стертий (стертий символ позначається через X). Цей канал обов'язково містить у собі модулятор і демодулятор: демодулятор видає в сумнівних випадках символ, що відповідає стиранню і відмінний від 0 і 1.

Приклад: нехай передана інформація 1000101 і помилка відбулася в другому символі.

Інформація на виході:

1) для двійкового симетричного каналу 1100101 (у даному випадку невідомо, є помилка в кодовому чи слові ні, і якщо є, те в якому символі);

2) для двійкового каналу, що стирає, 1x00101 (про помилку свідчить невизначеність у другому символі).

Задача виправлення помилок для двійкового симетричного каналу є більш складною в порівнянні з двійковим каналом, що стирає.

Надалі будемо розглядати тільки двійкові симетричні канали.

Класифікація перешкодостійких кодів

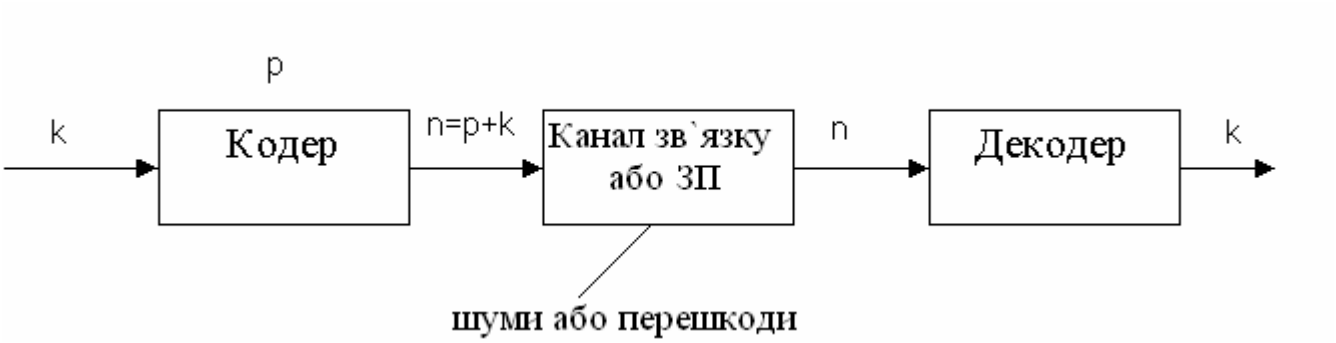
Перешкодостійкі коди - один з найбільш ефективних засобів забезпечення високої вірогідності передачі дискретної інформації. Історія розвитку перешкодостійкого кодування почалася в 1948р. публікацією знаменитої статті Клода Шенона. Шеннон показав, що для кожного каналу існує характеристика, вимірювана в бітах у секунду і називана *пропускною здатністю каналу* *C*. Якщо необхідна від системи зв'язку швидкість передачі інформації R (вимірювана в бітах у секунду) менше C (R<C), то, використовуючи коди, що виправляють помилки, для даного каналу можна побудувати таку систему зв'язку, що імовірність помилки на виході буде як завгодно мала. Таким чином, економічно вигідніше використовувати кодування, чим будувати занадто гарні канали. У теоремі Шеннона не говориться про те, як знайти підходящі коди, доведене лише їхнє існування.

Перешкодостійкі коди - коди, що дозволяють виявляти або виявляти і виправляти помилки, що виникають у результаті впливу перешкод. Перешкодостійке кодування забезпечується за рахунок уведення надмірності в кодові комбінації, тобто за рахунок того, що не всі символи в кодових комбінаціях використовуються для передачі інформації.

Усі перешкодостійкі коди можна розділити на два основних типи:

1. Блокові коди

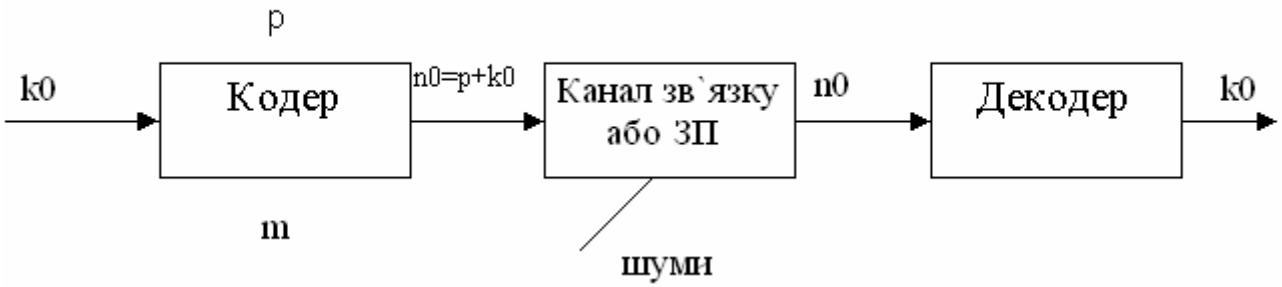
Вся інформаційна послідовність поділяється на блоки однакового розміру k біт (символів), і кожен блок обробляється окремо, тобто незалежно від інших блоків. Кодер додає к k p додаткових (перевірочних) символів. Задача декодера знайти або



виправити помилки.

n – довжина коду або значність коду;
k – кількість інформаційних символів;
p – кількість перевірочних символів.

2. Деревоподібні коди



Вся інформаційна послідовність також поділяється на блоки набагато меншого розміру k_0, що називаються кадрами (фреймами). Кожен інформаційний кадр кодується в залежності від m уже переданих інформаційних кадрів. Також додаються додаткові символи.

n_0 – кадр кодового слова;

k_0 – інформаційний кадр;

p – перевірочні символи.

Окремим випадком деревоподібних кодів є згортальні коди (коди Вайнера-Эша, Івадарі та ін.).

Для блокових і деревоподібних кодів існують різні види декодування. Для пошуку блокових кодів відповідні розділи математики були вже готові, тому кількість знайдених блокових кодів, у тому числі тих, що знайшли широке застосування, набагато більше, ніж деревоподібних.

Блокові коди

Блоковими кодами називаються звичайно (n, k) - коди, де n – кількість розрядів у кодовій комбінації (прийнято називати довжиною або значністю коду), k - кількість інформаційних розрядів. Ми будемо розглядати тільки двійкові перешкодостійкі коди, тобто коди з основою m = 2.

Символи кожного розряду кодової комбінації можуть приймати значення 0 або 1.

Кількість одиниць у кодовій комбінації називають ***вагою кодової комбінації*** і позначають w.

Приклад: кодова комбінація 1001001 характеризується значністю n = 7 і вагою w = 3.

Відстань за Хеммінгом характеризує ступінь відмінності будь-яких двох кодових комбінацій і позначається d. Вона виражається числом позицій або символів, у яких комбінації відрізняються одна від іншої, і визначається як вага посимвольної суми за модулем два цих комбінації.

Приклад: для визначення відстані за Хеммінгом між комбінаціями 10010010 і 11011000 потрібно підсумувати їх за модулем два.

$$\begin{array}{r} 10010010 \\ + \\ 11011000 \\ \hline 01001010^{[1]} \end{array}$$

Вага отриманої комбінації w = 3, тому відстань між вихідними комбінаціями дорівнює d = 3.

Перешкодостійкість кодування забезпечується за рахунок уведення надмірності в кодові комбінації. Це значить, що з n символів кодової комбінації для передачі інформації використовується k символів (k < n). Отже, із загального числа 2^n можливих кодових комбінацій для передачі інформації використовується тільки 2^k комбінацій (так називані *дозволені* комбінації, комбінації без помилок). Всі інші $2^n - 2^k$ кодові комбінації є забороненими.

Мінімальною кодовою відстанню за Хеммінгом називається найменша відстань за Хеммінгом між будь-якими парами дозволених кодових комбінацій і позначається d_{min}**.** Це дуже важлива характеристика коду, тому що саме вона характеризує його *коригувальні здібності* (під коригувальною здатністю коду мається на увазі здатність коду виявляти або виправляти помилки).

[1] У подальшому «+» - знак операції «сума за модулем два»

Зв'язок коригувальної здатності кодів з мінімальною відстанню за Хеммінгом

Уведемо позначення: t – кількість помилок, що виявляються;

s – кількість помилок, що виправляються.

Нехай необхідно побудувати код, що виявляє t і менше помилок. Для цього мінімальна відстань за Хеммінгом повинна задовольняти нерівності:

$$d_{min} \geq t+1,$$

тобто будь-який код, що виявляє t помилок, повинен мати мінімальну відстань за Хеммінгом більше або рівну t+1.

Для будь-якого коду, що виправляє s помилок, повинне виконуватися нерівність:

$$d_{min} \geq 2s+1$$

Для будь-якого коду, що виявляє t і виправляє s помилок, повинне виконуватися нерівність:

$$d_{min} \geq t+s+1$$

Приклад: розглянемо код зі значністю n = 3. Усі можливі комбінації такого коду мають вигляд:

A0	A1	A2	A3	A4	A5	A6	A7
000	001	010	011	100	101	110	111

Для того щоб код забезпечував виявлення однократних помилок (t=1), необхідно з усіх 8 можливих комбінацій вибрати в якості дозволених такі, відстань між якими було б не менш $d_{min} = 2$. Як приклад дозволених комбінацій у цьому випадку можна вибрати:

A0 = 000; A3 = 011; A5 = 101; A6 = 110.

Для виявлення дворазових помилок (t=2) $d_{min} = 3$. При цьому як приклад дозволених комбінацій можна вибрати:

A0 = 000; A7 = 111.

Нехай тепер необхідно побудувати код, що забезпечує виправлення одиночної помилки (s=1). Вибираємо в якості першої дозволеної комбінації A0 = 000. При наявності одиночних помилок комбінація A0 може перейти в A1 = 001, A2 = 010 або A4 = 100. Комбінації A1, A2 і A4 повинні бути забороненими. Тоді у випадку прийому однієї з них виноситься рішення, що передано комбінацію A0. Нехай у якості другої дозволеної комбінації вибирається комбінація, що відстоїть від першої на відстані $d_{min} = 2$, наприклад, A3= 011. Їй відповідає підмножина заборонених комбінацій A2 = 010, A1 = 001 і A7= 111. Таким чином, при прийомі A1 або A2 не можна однозначно установити, передана комбінація A1 або A4.

Якщо ж другою дозволеною комбінацію вибрати таку, що відстоїть від A0 на $d_{min} = 3$, тобто комбінацію A7 = 111, а підмножина заборонених комбінацій A3 = 011, A5 = 101 і A6 = 110, то в цьому випадку підмножини заборонених комбінацій не перетинаються. Отже, при $d_{min} = 3$ забезпечується усунення всіх одиночних помилок.

Визначення кількості перевірочних символів

Вихідними даними при розробці кодів можуть бути:

n – довжина коду, s (t) – кількість виправлюваних (або виявлюваних) помилок;

k – число інформаційних символів, s (t) – кількість виправлюваних (або виявлюваних) помилок.

Для побудови коду потрібно визначити кількість додаткових перевірочних символів p=n-k.

Для виявлення і виправлення одиночної помилки співвідношення між довжиною коду n, числом інформаційних розрядів k і числом коригувальних розрядів p повинне задовольняти наступним умовам:

$$2^p \geq n+1$$

$$2^k \geq \frac{2^n}{n+1} \quad ,$$

де n = k+p.

Для *практичних розрахунків* при визначенні p для кодів з мінімальною кодовою відстанню $d_{min} = 3$ зручно користуватися виразами, де цифра в індексі - число помилок, що виправляються, у дужках - що виявляються:

$$\mathbf{p_1(2) = [\ \log_2(n+1)\ ],}$$

якщо відомо довжину коду n, і

$$\mathbf{p_1(2) = [\log_2\{(k+1) + [\log_2 (k+1)]\}],}$$

якщо задане число інформаційних символів k (квадратні дужки означають округлення до найближчого **більшого** цілого).

Для *практичних розрахунків* при $s \geq 2$ використовують наступні формули:

S=2 $$p \geq \frac{n^2+n+1}{2}$$

S=3 $$p \geq \frac{n^3+n^2+n+1}{6}$$

У загальному випадку формула для виявлення числа перевірочних символів:

$$\log_2(C_n^S+C_n^{S-1}+\ldots+C_n^1+1) \leq p_S \leq \log_2(C_{n-1}^{2S-1}+C_{n-1}^{2S-2}+\ldots+n+1),$$

де ліва частина нерівності називається нижньою границею Хеммінга, а права – верхньою границею Варшамова – Гільберта.

Для практичних розрахунків можна використовувати формулу:

$$p \geq \frac{n^S+n^{S-1}+\ldots+n+1}{S!}$$

Коди з виявленням помилок

1. Код з парним числом одиниць

Код містить лише один надлишковий символ, що вибирається таким чином, щоб загальна кількість одиниць у кодовій комбінації була парною. Перевірка кодової комбінації виконується шляхом підсумовування за модулем два всіх його символів. Надмірність коду:

$$K_{надл} = p/n = 1/n.$$

Код дозволяє виявити всі помилки непарної кратності.

2. Код з подвоєнням елементів

Цей код характеризується введенням додаткових символів для кожного символу інформаційної частини комбінації, причому одиниця доповнюється нулем і перетворюється в 10, а нуль доповнюється одиницею і перетворюється в 01. Наприклад, комбінація 101 буде представлена у виді 100110. Показником перекручування коду буде поява в "парних" елементах сполучень виду 00 або 11. Надмірність коду

$$K_{надл} = 0,5.$$

Код дозволяє виявити всі помилки, за винятком випадку, коли мають місце дві помилки в "парних" елементах. Найбільш ймовірним видом помилок, що виявляються, є помилка в одному з "парних" елементів.

3. Інверсний код

В основу побудови інверсного коду покладений метод повторення вихідної кодової комбінації. Існує кілька різновидів такого коду. Одна з них полягає в наступному. У тих випадках, коли вихідна комбінація містить парне число одиниць, друга комбінація в точності відтворює вихідну. Якщо ж вихідна комбінація містить непарне число одиниць, то повторення виконується в інвертованому виді. Наприклад, комбінації 01010 і 11100 будуть представлені у виді 0101001010 і 1110000011. Надмірність коду

$$K_{надл} = 0,5.$$

Код дозволяє виявити практично всі помилки в кодовій комбінації, за винятком випадків, коли спотворяться два, чотири і т.д. елементи у вихідній комбінації і відповідні два, чотири і т.д. елементи в додатковій комбінації. З розглянутих кодів інверсний код володіє найбільшою перешкодостійкістю.

Коди, що виправляють помилки

Групові коди

Групові коди є блоковими, лінійними, систематичними.

Лінійними називаються такі коди, для яких посимвольна сума за модулем два двох дозволених комбінації також дає дозволену кодову комбінацію.

Систематичними кодами називаються такі коди, кодове слово яких складається з перших k інформаційних символів і наступних p перевірочних.

Матричне представлення групових кодів

Розрізняють два варіанти побудови групових кодів:

а) групові коди з мінімумом перевірочних символів;

б) групові коди з мінімальними апаратними витратами реалізації кодера і декодера.

Для обох варіантів будується породжувальна матриця P:

$$\mathbf{P_{(n,k)} = I\ H_p,}$$

Де I – одинична матриця (k×k);

H_p – перевірочна підматриця (k×p).

Існує дві умови для побудови перевірочної підматриці:

1. а) для групових кодів з мінімум перевірочних символів
кожен рядок перевірочної підматриці повинен мати вагу **не менш** d_{min}-1

б) для групових кодів з мінімальними апаратними витратами реалізації кодера і декодера
кожен рядок перевірочної підматриці повинен мати вагу, **рівну** d_{min}-1

2. Посимвольна сума будь-яких двох рядків перевірочної підматриці повинна мати вагу не менш d_{min}-2.

Для частки випадку групового коду, що виправляє одиночну помилку, остання умову означає, що всі рядки повинні бути різними: s=1, d_{min}=3, тобто рядки повинні відрізнятися хоча б у d_{min}-2=3-2=1 символі.

Крім перерахованих вище відмінностей, варіанти а) і б) будуть відрізнятися в реалізації дешифратора і декодера.

Приклад а): код з мінімумом перевірочних символів p.

Розглянемо груповий код, що виправляє одиночну помилку:

s=1 n=7 d_{min}= 3

$p \geq \log_2(n+1) = 3$

k = n-p = 4

Породжувальна матриця коду (7,4):

$$\mathbf{P}_{(7,4)} = \begin{vmatrix} a_1 & a_2 & a_3 & a_4 & b_1 & b_2 & b_3 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{vmatrix}$$

Оскільки кожен рядок перевірочної підматриці повинен містити не менш 2 одиниць, то можливі різні варіанти її побудови:

$$H_p = \begin{vmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{vmatrix} \sim \begin{vmatrix} 1 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{vmatrix} \sim \ \dots \ \sim \begin{vmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{vmatrix}$$

Однак, чим більше одиниць у перевірочній підматриці, тим більші апаратні витрати, тому побудову підматриці завжди починають з рядка, що містить мінімально припустиме число одиниць. Інші рядки добудовуються за принципом «одиниці, що біжить». У такому випадку не потрібно щоразу перевіряти при виборі чергового рядка відмінність його від уже записаних (особливо це актуально при великій довжині кодів). Наприклад,

$$\begin{vmatrix} 0 & 0 & \dots & 0 & 0 & 1 & 1 \\ 0 & 0 & \dots & 0 & 1 & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 0 & 1 & \dots & 0 & 0 & 0 & 1 \\ 1 & 0 & \dots & 0 & 0 & 0 & 1 \\ \dots & \dots & \dots & 0 & 1 & 1 & 0 \\ 0 & 0 & \dots & 1 & 0 & 0 & 1 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ 1 & 0 & \dots & 0 & 0 & 1 & 0 \\ 0 & 0 & \dots & 1 & 1 & 0 & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots \end{vmatrix}$$

Виходячи з цих положень, перший варіант побудови перевірочної підматриці є оптимальним.

Для побудови кодера і декодера головну роль грають перевірочні матриці:

$$\mathbf{H = Hp\ I,}$$

де Hp – транспонована перевірочна підматриця.

$$\mathbf{H} = \begin{vmatrix} a_1 & a_2 & a_3 & a_4 & b_1 & b_2 & b_3 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{vmatrix}$$

a1, a2, a3, a4 – інформаційні символи,

b1, b2, b3 – перевірочні символи.

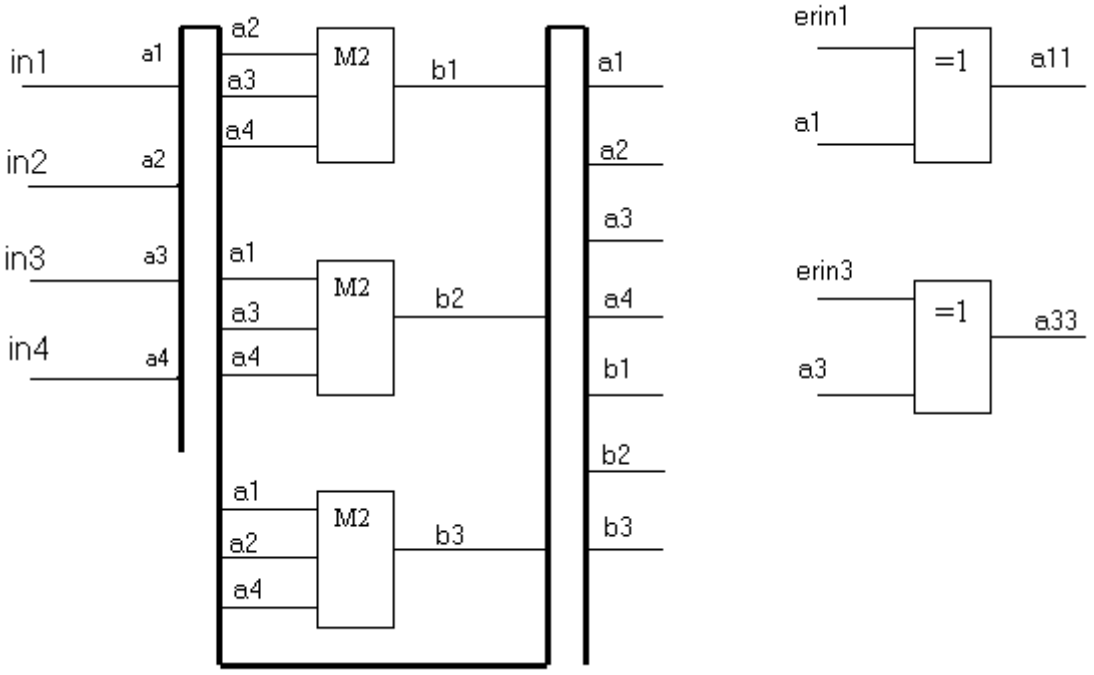
За перевірочною матрицею для побудови кодера визначаються вирази для перевірочних символів:

b1 = a2 + a3 + a4

b2 = a1 + a3 + a4

b3 = a1 + a2 + a4

У результаті отримана схема кодера:



У 1-му і 3-му символах виконується імітація помилки за допомогою сигналів erin1 і erin3 (a1→a11, a3→a33). Декодер аналізує інформаційні сигнали і порівнює них з перевірочними, у результаті чого формується ***синдром S.*** Якщо помилок нема, синдром дорівнює 0, і відмінний від 0, якщо є одиночні помилки.

S1 = b1 + a2 + a33 + a4

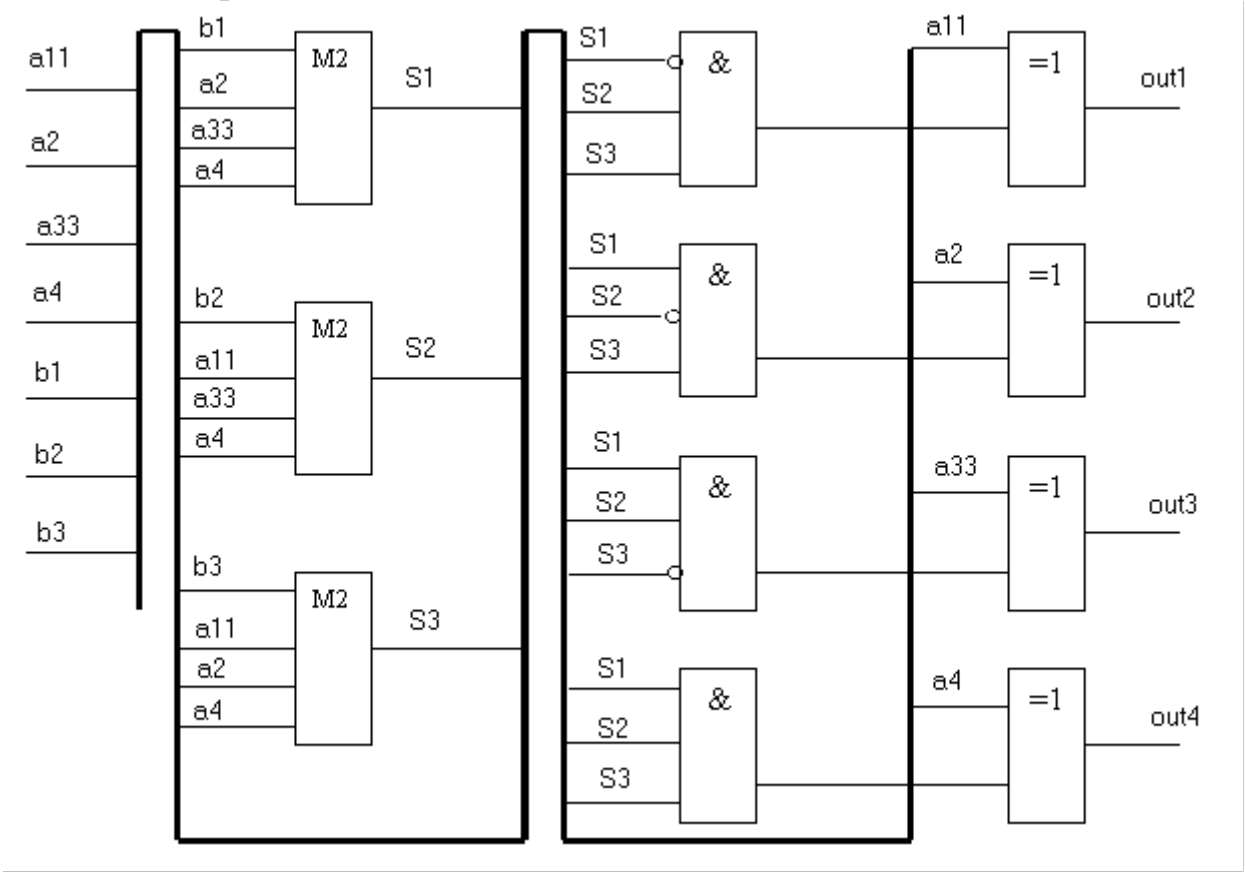
S2 = b2 + a11 + a33 + a4

S3 = b3 + a11 + a2 + a4

Синдром має наступну властивість: якщо була одиночна помилка в і-ому символі кодового слова, то синдром S дорівнює і-му стовпцеві перевірочної матриці H.

Наприклад, якщо помилка була в 1-ому інформаційному символі, то S = 011, якщо в 2-ом – S = 101 і т.д. (див. перевірочну матрицю H). Якщо помилка була подвійна, то (оскільки код лінійний) синдром дорівнює посимвольній сумі за модулем два відповідних стовпців перевірочної матриці. Нехай помилка була в 1-ом і в 3-ем символах: S = 101, що відповідає помилці в 2-му символі. Декодер виправляє 2-й символ, що не був помилковим, тобто додає в кодове слово ще одну помилку. Оскільки в 1-му і 3-му символах помилки не виправляються, на виході декодера буде три помилки.

Схема декодера:



Приклад б): код з мінімальними апаратними витратами.

Розглянемо груповий код, що виправляє одиночну помилку:

s=1 n=7 d_{min}= 3

$p \geq \log_2(n+1) = 3$

$k = n-p = 4$

Апаратні витрати визначаються числом входів і виходів схем кодера і декодера. Для зменшення апаратних витрат збільшуємо кількість перевірочних символів до p=4, оскільки забезпечити першу умову побудови перевірочної підматриці (кожен рядок повинен містити тільки дві одиниці і не більш) при p=3 у даному випадку не можна.

Породжувальна матриця:

$$\mathbf{P}_{(n,\,k)} = \begin{vmatrix} a_1 & a_2 & a_3 & a_4 & b_1 & b_2 & b_3 & b_4 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 \end{vmatrix}$$

Перевірочна матриця:

$$\mathbf{H} = \begin{vmatrix} a_1 & a_2 & a_3 & a_4 & b_1 & b_2 & b_3 & b_4 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{vmatrix}$$

Вираз для перевірочних символів:
b1 = a4
b2 = a2 + a3
b3 = a1 + a3
b4 = a1 + a2 + a4 a1 -> a11, a3 -> a33
Вираз для розрядів синдрому:
S1 = b1 + a4
S2 = b2 + a2 + a33
S3 = b3 + a11 + a33
S4 = b4 + a11 + a2 + a4

Вибравши p=4, отримуємо, що кожен стовпець перевірочної матриці, що відповідає інформаційному символові, містить рівно дві одиниці, і усі вони різні, тому при реалізації дешифратора можна використовувати двовхідні елементи “і”.

Схема кодера:

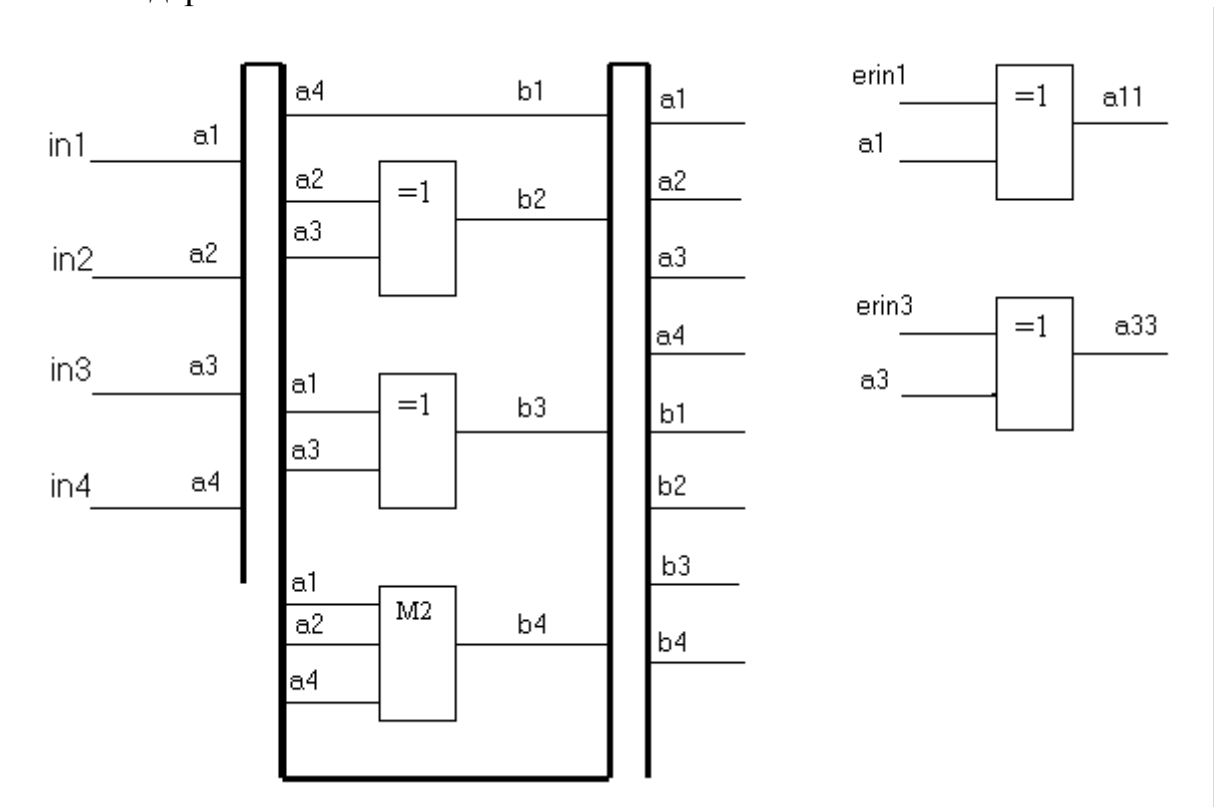
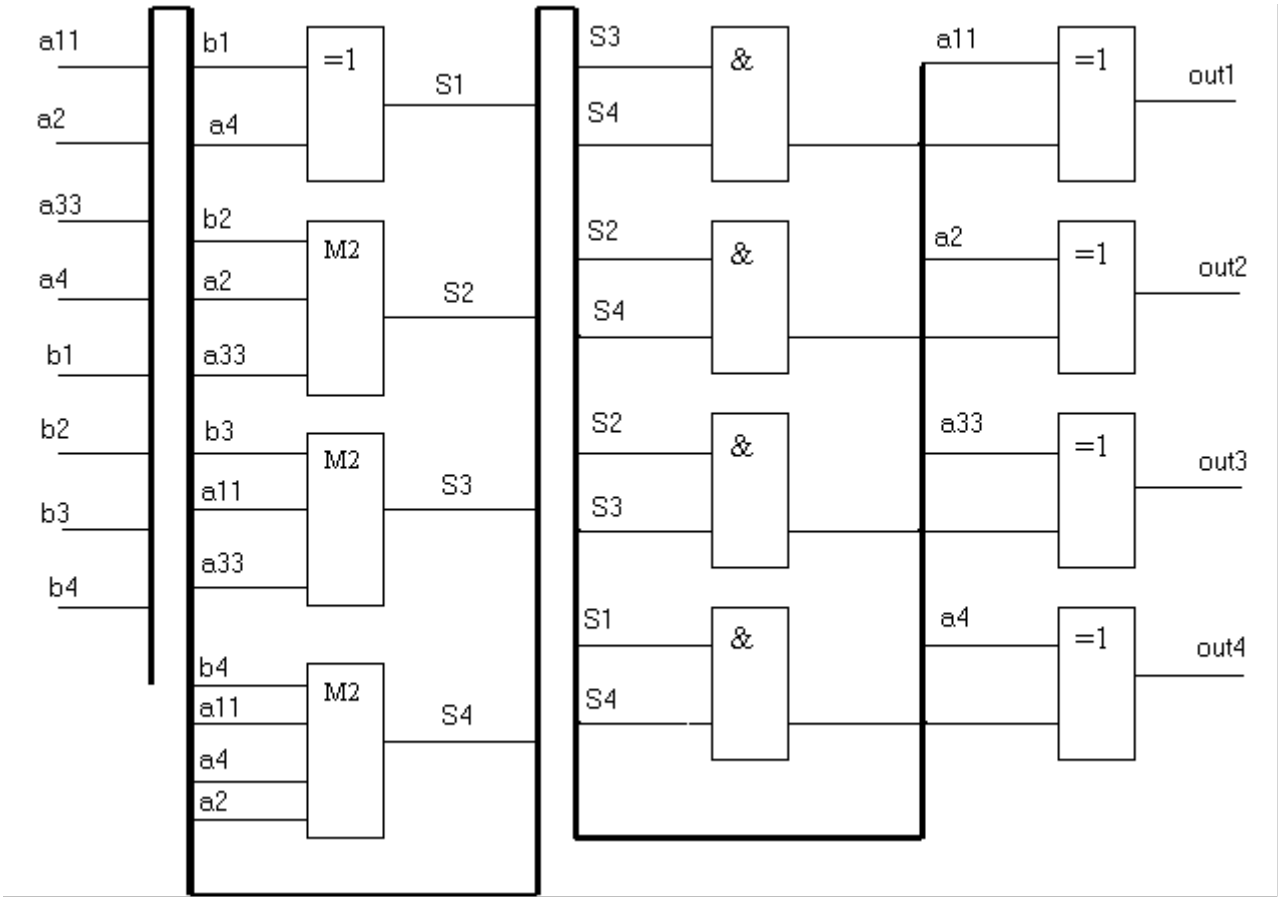


Схема декодера:



Коди Хеммінга

Код Хеммінга являє собою один з найважливіших класів лінійних кодів, що знайшли широке застосування на практиці. До цих кодів звичайно відносять коди з виправленням одиночних помилок і коди з виправленням одиночних і виявленням дворазових помилок. Особливість цих кодів – можливість використання в декодері стандартного дешифратора.

Коди Хеммінга, що виправляють одиночну помилку ($d_{min}= 3$)

Для кодів **Хеммінга**, що виправляють одиночну помилку, відразу будується перевірочна матриця, що містить p рядків і n стовпців. Кожен стовпець перевірочної матриці дорівнює номерові позиції стовпця в двійковому виді. Якщо стовпець містить одну одиницю, він відповідає перевірочному символові. Всі інші стовпці відповідають інформаційним символам.

Приклад: вихідні дані для побудови коду **Хеммінга**

$s=1,\ d_{min}= 3,\ k = 4$

$p \geq [\log_2\{(k + 1) + [\log_2 (k + 1)]\}],\ p = 3$

$n = k+p = 7$

Будуємо перевірочну матрицю для коду (7,4):

$$\mathbf{H} = \begin{vmatrix} b_1 & b_2 & a_3 & b_4 & a_5 & a_6 & a_7 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{vmatrix}$$

Оскільки кожен стовпець, що відповідає перевірочному символові, має тільки по одній одиниці, то для складання виразів для перевірочних символів b розглядається той рядок, у якому знаходиться ця одиниця. Відповідний перевірочний символ визначається як сума за модулем два тих інформаційних символів a, що також мають одиницю в цьому рядку. Для даного прикладу

b1 = a3 + a5 + a7

b2 = a3 + a6 + a7

b4 = a5 + a6 + a7

Припустимо, що помилки імітуються в 3-м і 5-м символах:

a3 -> a33, a5 -> a55.

Аналогічно груповим кодам складаються вирази для розрядів синдрому

S0 = b1 + a33 + a55 + a7

S1 = b2 + a33 + a6 + a7

S2 = b4 + a55 + a6 + a7

Для реалізації декодера застосовується стандартний дешифратор.

Схема кодера:

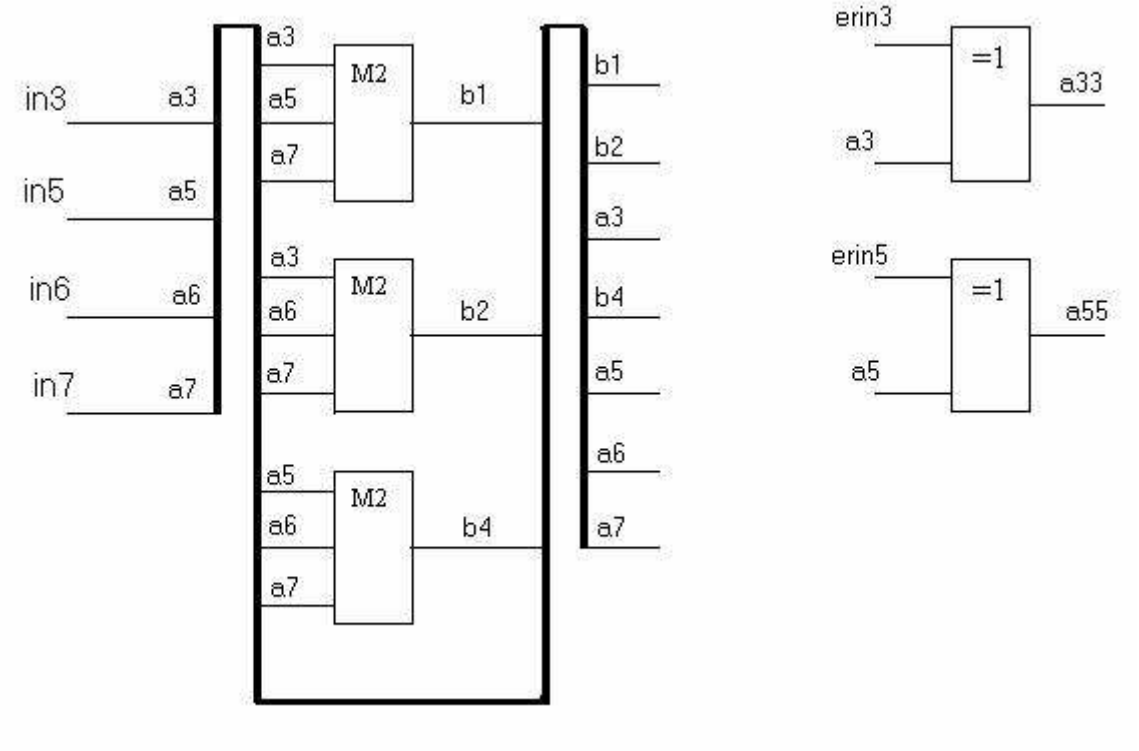
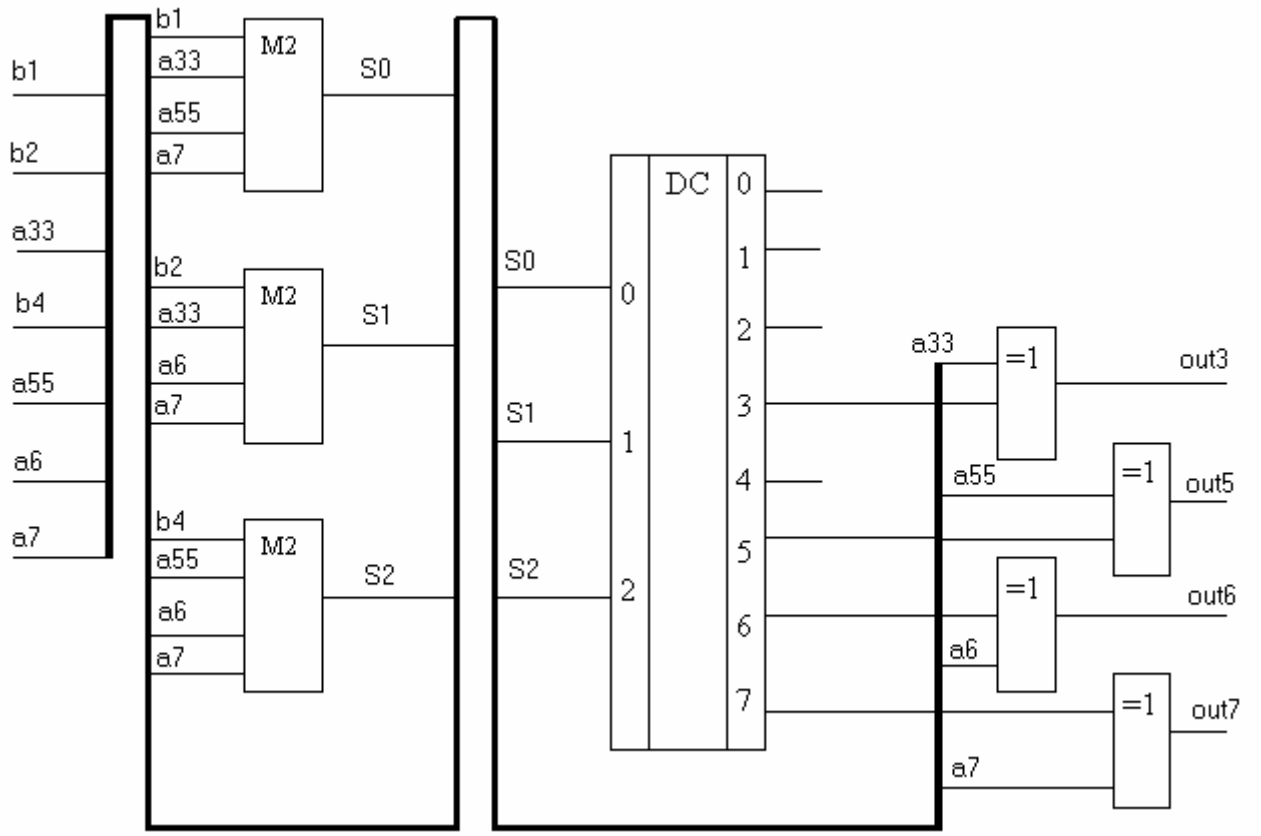


Схема декодера:



Оскільки інформаційні символи займають тільки позиції 3,5,6,7 у матриці H, то виходи 0, 1, 2, 4 дешифратора не використовуються для виявлення помилки.

У випадку подвійної помилки, так само як і в групових кодах, на виході може бути не дві, а три помилки. Наприклад, якщо помилки були в 3-м і 5-м символах, синдром буде дорівнювати лінійній комбінації чисел 3 і 5 у двійковому представленні: S=110, а цей синдром відповідає 6-му символові, що не був перекручений. Декодер, виправляючи 6-й символ, додає в кодове слово 3-ю помилку.

Коди Хеммінга, що виявляють подвійні і виправляють одиночні помилки ($d_{min}= 4$)

Код Хеммінга з кодовою відстанню d min = 4 виходить шляхом додавання до коду Хеммінга з d min = 3 перевірочного символу, що являє собою результат підсумовування за модулем два всіх символів кодової комбінації. Іншими словами код Хеммінга, що виправляє одиночні помилки, доповнюється кодом за паритетом. Перевірочна матриця при цьому доповнюється рядком із всіх одиниць і стовпцем із усіх нулів.

Приклад: вихідні дані для побудови коду

$S=1,\ d_{min}= 4,\ t=2,\ k=4$

$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}]+1 = 4$

$n = k + p = 8$

Перевірочна матриця для коду (8,4)

$$\mathbf{H} = \begin{vmatrix} b_1 & b_2 & a_3 & b_4 & a_5 & a_6 & a_7 & b \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{vmatrix}$$

Вирази для перевірочних символів

b1 = a3 + a5 + a7

b2 = a3 + a6 + a7

b4 = a5 + a6 + a7

b = b1 + b2 + a3 + b4 + a5 + a6 + a7

a3 -> a33, a5 -> a55

Вирази для розрядів синдрому

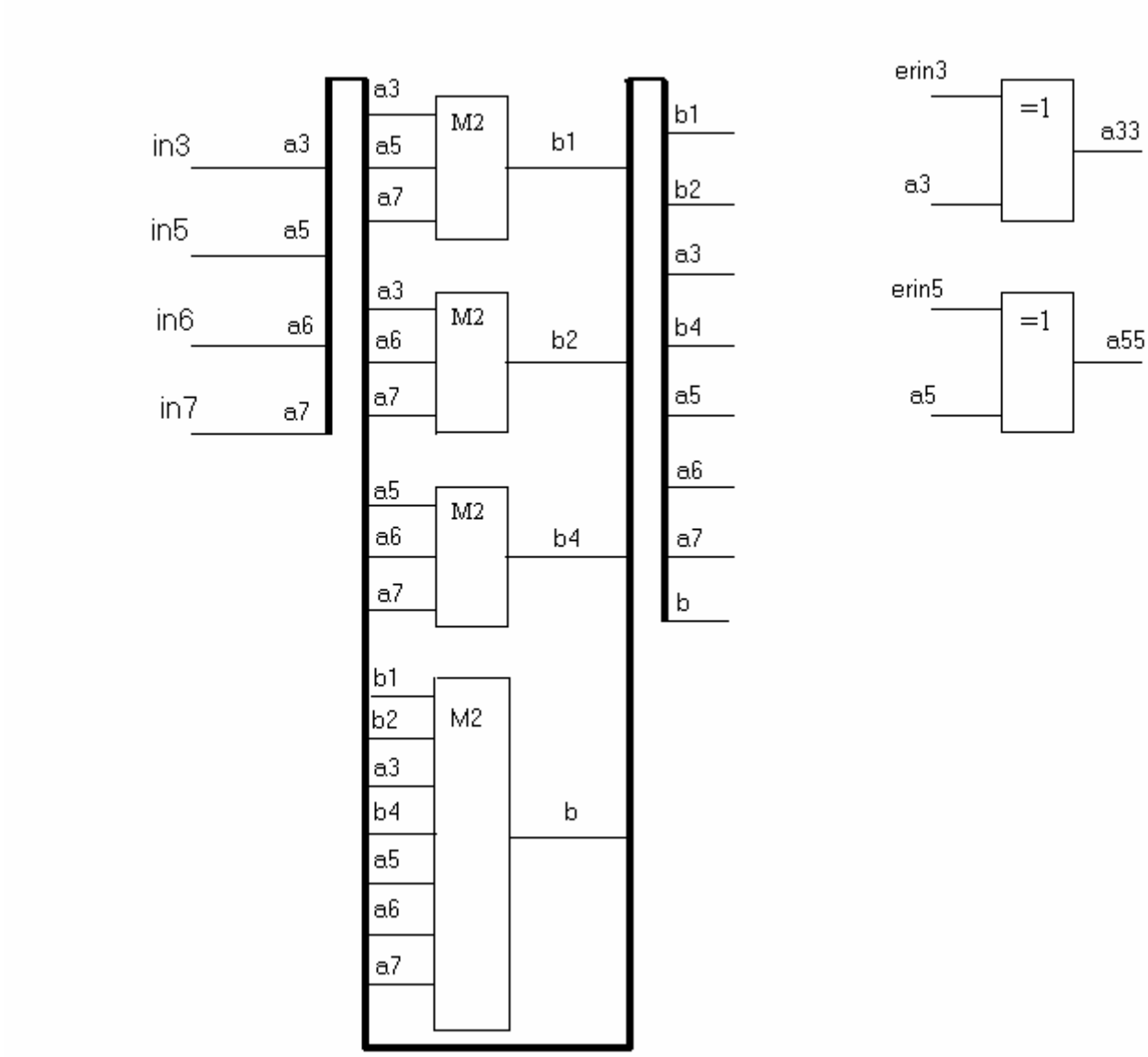
S0 = b1 + a33 + a55 + a7

S1 = b2 + a33 + a6 + a7

S2 = b4 + a55 + a6 + a7

S = b + b1 + b2 + a33 + b4 + a55 + a6 + a7

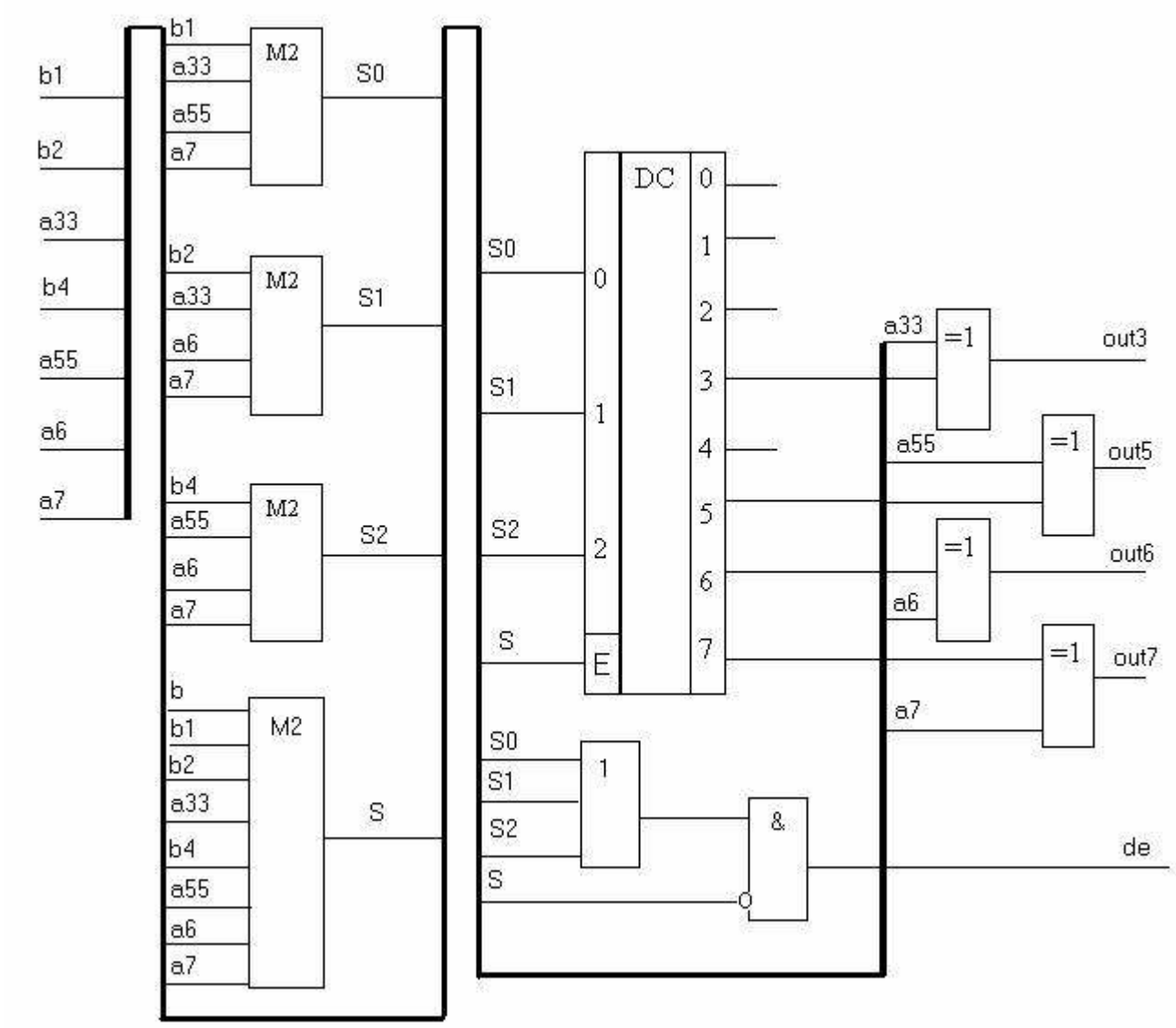
Схема кодера



Для того, щоб у випадку подвійної помилки в інформаційних символах виключити появу в кодовому слові третьої помилки, у даному випадку доцільно використовувати дешифратор із входом вибірки E. На цей вхід подається додатковий розряд синдрому s, якому відповідає останній рядок перевірочної матриці. У випадку одиночної помилки s = 1 (код за паритетом) декодер працює аналогічно попередньому прикладові – виправляє помилку. У випадку подвійної помилки, s = 0 і виправлення заборонене – додавання помилки не буде.

Для формування сигналу de (double error) про подвійну помилку використовувати просто сигнал s не можна, оскільки він дорівнює нулеві не тільки у випадку подвійної помилки, але і при відсутності помилок. З огляду на те, що у випадку подвійної помилки інші розряди синдрому S будуть відмінні від нуля (синдром дорівнює лінійної комбінації двох стовпців перевірочної матриці, а всі стовпці відмінні друг від друга) і s =0, можна розрізнити ситуації відсутності помилок (S=0 і s=0) і наявності подвійної помилки в кодовому слові.

Схема декодера:



Циклічні коди

Ці коди знайшли широке застосування завдяки простій апаратній реалізації і високим коригувальним здібностям. Свою назву ці коди одержали через їхню властивість: будь-яке циклічний зсув дозволеного кодового слова приводить до дозволеного кодового слова.

Наприклад, якщо кодове слово $C_1C_2C_3$…....$C_{n-1}C_n$ дозволене, тоді кодове слово $C_nC_1C_2C_3$…....C_{n-1} також дозволене.

Для побудови й аналізу циклічних кодів використовується поліноміальне представлення двійкових послідовностей.

Представлення двійкових кодів у виді поліномів

Поліномом називається математичне вираження

$$f(x)= f_n*x^n + f_{n-1}*x^{n-1} + \ldots + f_1*x + f_0,$$

де $f_i \in \{0,1\}$ (двійковий випадок), x – псевдозмінна (фіктивна).

Приклад:

	6	5	4	3	2	1	0
A:	1	0	1	0	1	1	1

$A(x) = x^6 + x^4 + x^2 + x + 1$

Операції над поліномами

1. Додавання/ Віднімання

Для двійкового випадку дані операції еквівалентні.

a	b	c
0	0	0
0	1	1
1	0	1
1	1	0

Приклад:

$$\begin{array}{l} A(x)=x^6+x^4+x^2+x+1 \\ \pm \\ B(x)=x^7+x^5+x^4+x^3+x+1 \\ \hline C(x)=x^7+x^6+x^5+x^3+x^2 \end{array}$$

2. Множення поліномів

$$A(x)=x^5+x^2+1$$
$$*$$
$$B(x)=x^2+x+1$$
$$x^5+x^2+1$$
$$+\ x^6+x^3+x$$
$$+\ x^7+x^4+x^2$$
$$C(x)=x^7+x^6+x^5+x^4+x^3+x+1$$

3. Ділення поліномів

$C(x) = x^7 + x^6 + x^5 + x^4 + x^3 + x + 1$

$A(x) = x^5 + x^2 + 1$

$$x^7+x^6+x^5+x^4+x^3+x+1 \mid x^5+x^2+1$$
$$-x^7+x^4+x^2 \quad \mid x^2+x+1$$
$$x^6+x^5+x^3+x^2+x+1$$
$$-x^6+x^3+x$$
$$x^5+x^2+1$$
$$-x^5+x^2+1$$
$$0$$

$R_{A(x)}[C(x)] = 0$ – залишок від ділення полінома C(x) на поліном A(x)

$A(x) = x^5 + x^2 + 1$

$B(x) = x^3 + 1$

$$x^5+x^2+1 \mid x^3+1$$
$$-x^5+x^2 \quad \mid x^2$$
$$1$$

2 1 0 ↙

R= 001

Залишок від ділення повинен бути менше степеня дільника B(x).

Степенем полінома називається максимальний степінь псевдозмінних з ненульовими коефіцієнтами.

deg A(x) = 5; deg B(x) = 3.

Побудова циклічних кодів

Центральну роль при побудові циклічних кодів грають так називані ***породжувальні поліноми (утворюючі поліноми)***. Для циклічних кодів Хеммінга породжувальними використовуються так називані ***незвідні поліноми***. Незвідним поліномом називається такий поліном, що не може бути представлений у виді добутку поліномів нижчих степенів (аналогія з простими числами). Незвідні поліноми вибираються з таблиць.

Кількість перевірочних символів циклічного коду дорівнює степеню породжувального полінома і, навпаки, степінь породжувального полінома дорівнює числу перевірочних символів.

Циклічні коди Хеммінга

Розрізняють систематичні і несистематичні циклічні коди.

Основна ідея циклічного кодування полягає в тому, що дозволене кодове слово повинне ділитися на породжувальний поліном, без залишку.

Залишок від ділення прийнятого декодером кодового слова на породжувальний поліном, виступає в ролі синдрому. Якщо залишок дорівнює нулю, декодер вважає, що помилок не було. У випадку ненульового синдрому в залежності від його виду декодер виконує виправлення помилок.

1. Побудова систематичного циклічного коду Хеммінга

Приклад. Нехай, наприклад, інформаційні символи дорівнюють комбінації 1101.

У цьому випадку k=4. Для кодів Хеммінга кількість помилок, що виправляються, дорівнює s=1, тоді $d_{min}= 3$.

Поліноміальне представлення інформаційних символів має вигляд

$A(x) = x^3 + x^2 + 1$.

По відомій формулі визначаємо кількість перевірочних символів:

$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}] = 3$.

Довжина коду:

$n = k + p = 7$.

$\deg K(x) = p = 3$ – степінь породжувального полінома.

$K(x) = x^3 + x + 1$ (вибираємо з таблиці незвідних поліномів).

Кодер систематичного циклічного коду Хеммінга повинен виконувати операцію множення інформаційної послідовності на поліном x^p і ділення на породжувальний поліном. Отриманий залишок від ділення являє собою перевірочні символи.

$$\mathbf{A(x)* x^p + R_{K(x)}[A(x)*x^p]}$$

$A(x)*x^p = (x^3 + x^2 + 1)*x^3 = x^6 + x^5 + x^3$ | $x^3 + x + 1$

$-\,x^6 + x^4 + x^3$ | $x^3 + x^2 + x + 1$

$x^5 + x^4$

$-\,x^5 + x^3 + x^2$

$x^4 + x^3 + x^2$

$-\,x^4 + x^2 + x$

$x^3 + x$

$-\,x^3 + x + 1$

1 → 001 (2 1 0)

Одержали систематичний код $x^6 + x^5 + x^3 + 1 = 1101\ 001$, де перша частина 1101 інформаційна, а друга 001 – перевірочні символи.

Отже, отримане кодове слово повинне ділитися на породжувальний поліном, без залишку:

$x^6 + x^5 + x^3 + 1$ | $x^3 + x + 1$

$-\,x^6 + x^4 + x^3$ | $x^3 + x^2 + x + 1$

$x^5 + x^4 + 1$

$-\,x^5 + x^3 + x^2$

$x^4 + x^3 + x^2 + 1$

$-\,x^4 + x^2 + x$

$x^3 + x + 1$

$-\,x^3 + x + 1$

0

Декодер систематичного циклічного коду ділить прийняте кодове слово на породжувальний поліном, і в залежності від виду синдрому (залишку від ділення) виправляє кодове слово.

2. Побудова несистематичного циклічного коду Хеммінга

Приклад: Розглянемо побудову несистематичного коду Хеммінга при тих же початкових умовах.

Нехай, наприклад, інформаційні символи дорівнюють комбінації 1101.

У цьому випадку k=4. Для кодів Хеммінга кількість помилок, що виправляються, дорівнює s=1, тоді $d_{min} = 3$.

Поліноміальне представлення інформаційних символів має вигляд
$A(x) = x^3 + x^2 + 1$.

По відомій формулі визначаємо кількість перевірочних символів:

$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}] = 3$.

Довжина коду:

$n = k + p = 7$.

deg K(x) = p = 3 – степінь породжувального полінома.
K(x) = $x^3 + x + 1$ (вибираємо з таблиці незвідних поліномів).

Кодер несистематичного циклічного коду Хеммінга повинен виконувати операцію множення інформаційних символів на породжувальний поліном. У підсумку виходить кодове слово

A(x)* K(x)

$$A(x)*K(x)=(x^3+x^2+1)*(x^3+x+1)=x^6+x^4+x^3+x^5+x^3+x^2+x^3+x+1=x^6+x^5+x^4+x^3+x^2+x+1$$

6 5 4 3 2 1 0
1 1 1 1 1 1 1

Одержали кодове слово несистематичного коду $x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$

У цьому кодовому слові не можна визначити, де знаходяться інформаційні символи, а де перевірочні. Зокрема, в інформаційних символах присутній один символ 0, а в кодовому слові немає жодного нуля.

Декодер несистематичного циклічного коду Хеммінга ділить прийняте кодове слово на породжувальний поліном. По виду синдрому (залишку від ділення) виправляє кодове слово, а потім виправлене кодове слово ще раз ділить на породжувальний поліном, для відновлення інформаційних символів.

Матричне представлення циклічних кодів Хеммінга

1. Систематичний циклічний код

Приклад: Нехай, наприклад, кількість інформаційних символів k=4.
Для кодів Хеммінга кількість помилок, що виправляються, дорівнює s=1, тоді dmin= 3.
По відомій формулі визначаємо кількість перевірочних символів:
$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}] = 3$.
Довжина коду:
n = k + p = 7.
deg K(x) = p = 3 – степінь породжувального полінома.
K(x) = $x^3 + x + 1$ (вибираємо з таблиці незвідних поліномів).

Породжувальна матриця систематичного коду Хеммінга:

$$\mathbf{P_{(n,k)} = I\ H_p}$$

де I – одинична матриця, має k рядків і k стовпців;

H_p – перевірочна підматриця, що являє собою матрицю з k рядків і p стовпців. Кожен рядок являє собою перевірочні символи для відповідної комбінації інформаційних символів одиничної матриці.

Породжувальна матриця дозволяє одержати k дозволених комбінацій коду. Інші дозволені комбінації виходять підсумовуванням за модулем два рядків породжувальної матриці у всіх можливих сполученнях.

Визначення перевірочних символів для побудови породжувальної матриці:

1. k = 0001

$$\begin{array}{r|l} x^3 & x^3 + x + 1 \\ \underline{-x^3 + x + 1} & \\ x + 1 & 1 \end{array}$$

2 1 0

p = 011

2. k = 0010

$$\begin{array}{r|l} x^4 & x^3 + x + 1 \\ \underline{-x^4 + x^2 + x} & \\ x^2 + x & x \end{array}$$

2 1 0

p = 110

3. k = 0100

$$\begin{array}{r|l} x^5 & x^3 + x + 1 \\ \underline{-x^5 + x^3 + x^2} & \\ x^3 + x^2 & x^2 + 1 \\ \underline{-x^3 + x + 1} & \\ x^2 + x + 1 & \end{array}$$

2 1 0

p = 111

4. k = 1000

$$
\begin{array}{l|l}
x^6 & x^3 + x + 1 \\
\underline{- x^6 + x^4 + x^3} & \\
x^4 + x^3 & x^3 + x + 1 \\
\underline{- x^4 + x^2 + x} & \\
x^3 + x^2 + x & \\
\underline{- x^3 + x + 1} & \\
x^2 + 1 &
\end{array}
$$

$$p = 101^{2\,1\,0}$$

Породжувальна матриця для систематичного коду Хемминга (7,4):

$$P = \begin{array}{c} {\scriptstyle 3\ 2\ 1\ 0} \\ \begin{matrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{matrix} \end{array}$$

Для конкретних інформаційних символів для визначення кодового слова необхідно посимвольно скласти за модулем два рядки породжувальної матриці номери яких дорівнюють номерам позицій одиниць у комбінації інформаційних символів. Наприклад, для інформаційних символів 1101 необхідно скласти перший, другий і четвертий рядки. Одержуємо кодове слово 1101 001.

2. Несистематичний циклічний код

Приклад: Нехай, наприклад, кількість інформаційних символів k=4.

Для кодів Хеммінга кількість помилок, що виправляються, дорівнює s=1, тоді dmin= 3.

По відомій формулі визначаємо кількість перевірочних символів:

$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}] = 3.$

Довжина коду:

$n = k + p = 7.$

$\deg K(x) = p = 3$ – степінь породжувального полінома.

$K(x) = x^3 + x + 1$ (вибираємо з таблиці незвідних поліномів).

Породжувальна матриця будується аналогічно систематичному циклічному кодові, але символи кодового слова визначаються відповідно до формули для несистематичного коду.

Визначення символів k дозволених кодових слів для несистематичного циклічного коду (7,4).

Одинична матриця:

$$I = \begin{matrix} 3 & 2 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{matrix}$$

1. k = 1000

Множимо перший рядок одиничної матриці на породжувальний поліном:

$x^3 * (x^3 + x + 1) = x^6 + x^4 + x^3 = 1011000$

2. k = 0100

Множимо другий рядок одиничної матриці на породжувальний поліном:

$x^2 * (x^3 + x + 1) = x^5 + x^3 + x^2 = 0101100$

3. k = 0010

Множимо третій рядок одиничної матриці на породжувальний поліном:

$x * (x^3 + x + 1) = x^4 + x^2 + x = 0010110$

3. k = 0001

Множимо четвертий рядок одиничної матриці на породжувальний поліном:

$1 * (x^3 + x + 1) = x^3 + x + 1 = 0001011$

Породжувальна матриця для несистематичного коду Хеммінга (7,4):

$$P = \begin{matrix} 3 & 2 & 1 & 0 & & & \\ 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{matrix}$$

Для конкретних інформаційних символів для визначення кодового слова необхідно посимвольно скласти за модулем два рядки породжувальної матриці, номера яких дорівнюють номерам позицій одиниць у комбінації інформаційних символів. Наприклад, для інформаційних символів 1101 необхідно скласти перший, другий і четвертий рядки. Одержуємо кодове слово 1111111.

Схемна реалізація циклічного кодування

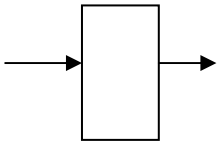
Для схемної реалізації циклічного кодування використовуються лінійні перемикальні схеми або, по іншій термінології, цифрові фільтри.

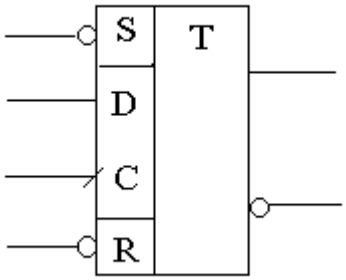
Лінійні перемикальні схеми (ЛПС)

Для ЛПС визначені три види елементів.

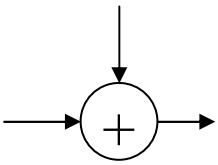
Елемент ЛПС	Аналог обчислювальної техніки

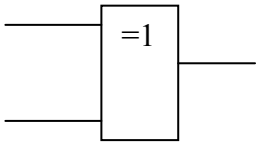
1. Елемент пам'яті або елемент затримки, має один вхід і один вихід:



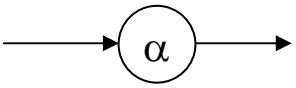


2. Суматор за модулем два (для двійкового випадку), має два входи й один вихід:





3. Помножувач на константу:



$\alpha = 1$ – зв'язок присутній
$\alpha = 0$ – зв'язок відсутній

У ЛПС допускається до виходу елементів підключати будь-яку кількість входів. Однак забороняється з'єднувати будь-які два виходи. У цифрових схемах існує поняття „навантажувальна здатність", з огляду на яку до виходу елемента не можна підключати необмежену кількість входів. Це варто враховувати при розробці принципових схем.

ЛПС із скінченним числом станів називаються схеми, що містять скінченне число елементів пам'яті (затримки), суматорів та помножувачів на константу, з'єдна-

них будь-яким припустимим способом. Розглянуті схеми надалі будуть використовуватися для схемної реалізації циклічних кодів.

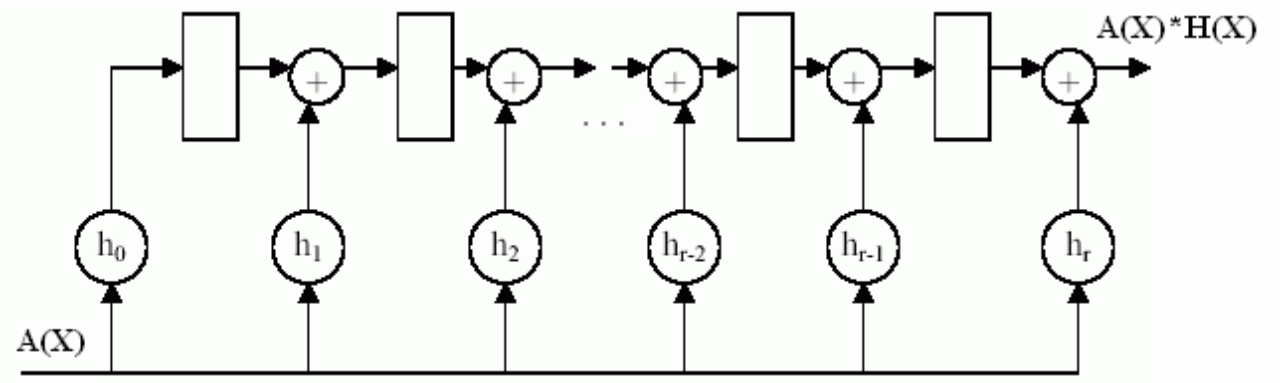
Будемо вважати, що на входи ЛПС надходить інформація в поліноміальному представленні, починаючи зі старших степенів. При цьому на виході ЛПС результат виходить, також починаючи зі старших степенів.

Схема для множення поліномів

Нехай є поліном-константа:

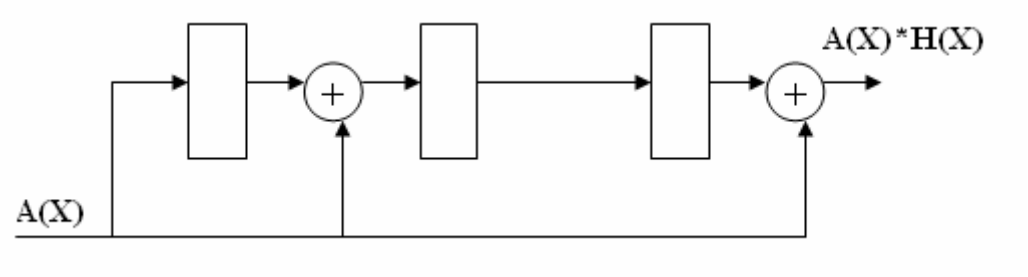
$$H(x) = h_r*x^r + h_{r-1}*x^{r-1} + \ldots + h_1*x + h_0$$

У загальному виді схема множення поліномів A(x) і H(x) буде містити r елементів пам'яті, r суматорів.



Для одержання результату на виході необхідна кількість тактів роботи схеми, рівна степені результату добутку плюс 1 (враховується нумерація степенів, починаючи з нуля), тобто сумі степенів поліномів A(x) і H(x) плюс 1.

Приклад. Множення поліномів $A(x) = x^3 + x^2 + 1$ і $H(x) = x^3 + x + 1$



Поліном $A(x) = x^3 + x^2 + 1$, тобто в двійковому виді – 1101. Початковий стан всіх елементів пам'яті - 000. Для одержання результату на виході необхідна кількість тактів роботи схеми, рівна сумі степенів поліномів A(x) і H(x) плюс 1, тобто - 7.Таким чином, “покрокова” робота схеми:

A(X)	0	0	0	A(X)*H(X)
1	1	1	0	1
1	1	0	1	1
0	0	1	0	1
1	1	1	1	1
0	0	1	1	1
0	0	0	1	1
0	0	0	0	1

На виході одержуємо комбінацію 11111111 або в поліноміальному представленні $x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$, правильний добуток:

$$(x^3 + x^2 + 1)*(x^3 + x + 1) = x^6 + x^4 + x^3 + x^5 + x^3 + x^2 + x^3 + x + 1 = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$$

6 5 4 3 2 1 0
1 1 1 1 1 1 1

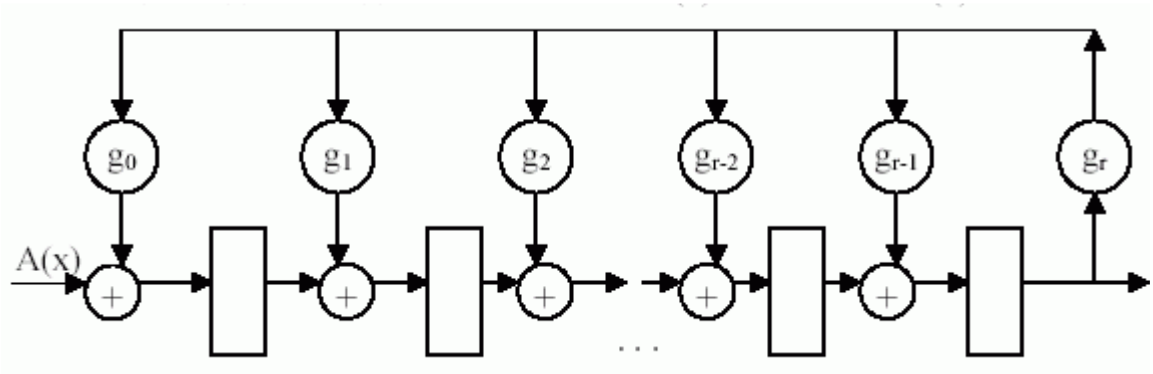
Таким чином, за допомогою даної схеми множення поліномів реалізується кодер для несистематичного циклічного коду Хеммінга.

Схема для ділення поліномів

Нехай є поліном константа:

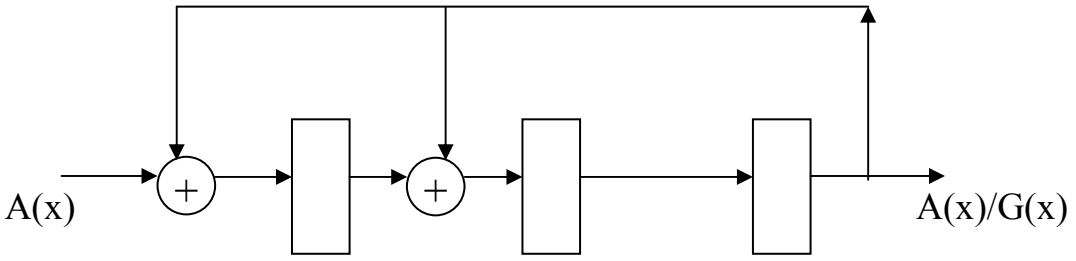
$$G(x) = g_r*x^r + g_{r-1}*x^{r-1} + \ldots + g_1*x + g_0$$

У загальному виді схема ділення полінома A(x) на поліном G(x):



По закінченні ділення елементи пам'яті містять залишок ділення $R_{G(x)}[A(x)]$ у наступному порядку (зліва направо) $1, X, X^2, \ldots, X^{r-2}, X^{r-1}$.

Приклад. Ділення полінома $A(x) = x^6 + x^5 + x^3$ на $G(x) = x^3 + x + 1$



- порядок проходження елементів залишку.

Ілюстрація ділення за допомогою схеми:

A(X)	0	0	0	A(X)/G(X)
1	1	0	0	0
1	1	1	0	0
0	0	1	1	1
1	0	1	1	1
0	1	1	1	1
0	1	0	1	1
0	1	0	0	0

Перевірка звичайним діленням:

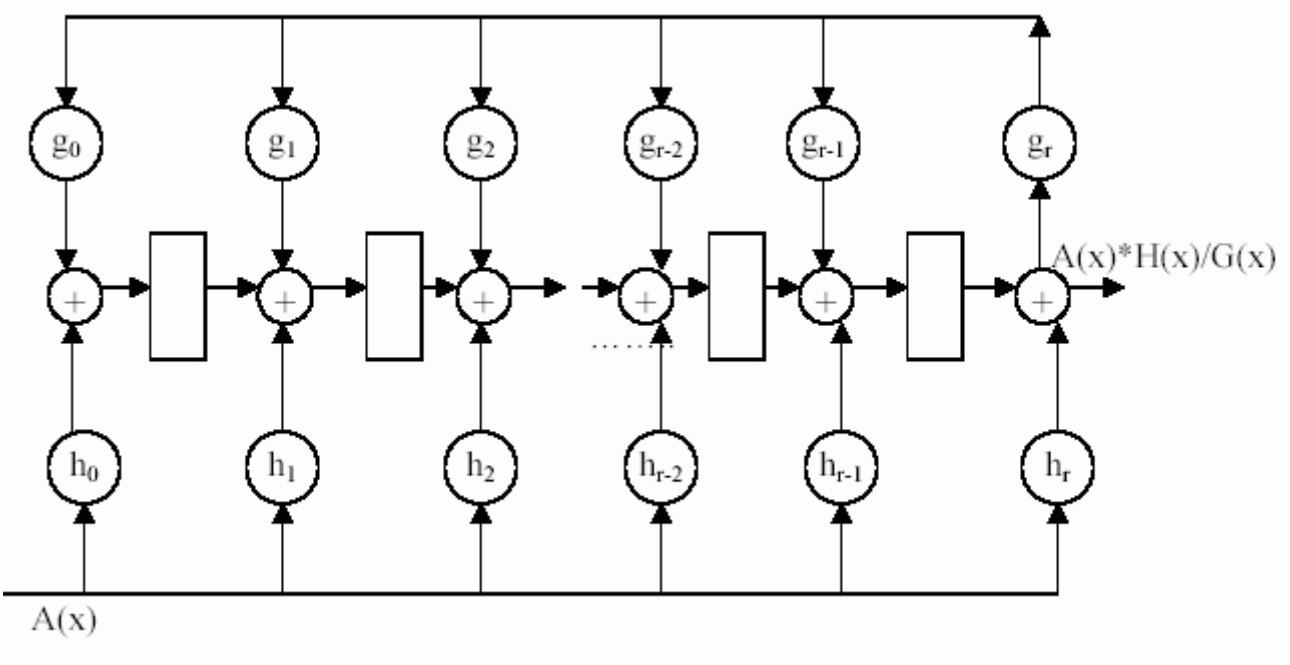
$$
\begin{array}{l|l}
x^6 + x^5 + x^3 & x^3 + x + 1 \\
\underline{- x^6 + x^4 + x^3} & \\
x^5 + x^4 & x^3 + x^2 + x + 1 \\
\underline{- x^5 + x^3 + x^2} & \\
x^4 + x^3 + x^2 & \\
\underline{- x^4 + x^2 + x} & \\
x^3 + x & \\
\underline{- x^3 + x + 1} & \\
1 & \longrightarrow 001
\end{array}
\qquad 2\,1\,0
$$

Таким чином, частка і залишок після ділення за допомогою схеми збігаються з часткою і залишком після звичайного ділення.

Схема для множення і ділення поліномів

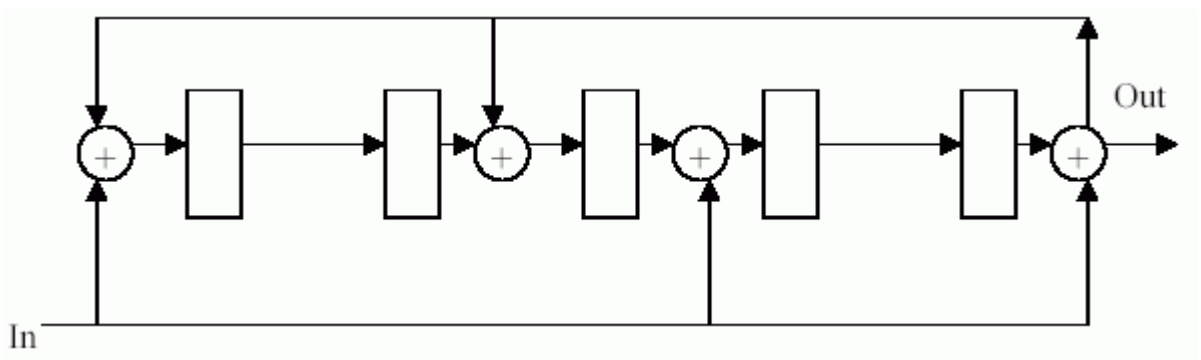
Нехай $H(x) = h_r*x^r + h_{r-1}*x^{r-1} + \ldots + h_1*x + h_0$ – поліном, на який помножується інформаційна послідовність, $G(x) = g_r*x^r + g_{r-1}*x^{r-1} + \ldots + g_1*x + g_0$ - поліном, на який ділиться інформаційна послідовність.

У загальному випадку схема для множення і ділення поліномів має вигляд:



По закінченню роботи схеми елементи пам'яті містять залишок від ділення $R_{G(x)}[A(x)*H(x)]$.

Приклад. Нехай $H(x) = x^5 + x^3 + 1$ і $G(x) = x^5 + x^2 + 1$.



Отже, розглянуті основні функціональні схеми, використовувані для схемної реалізації циклічних кодів.

Схеми кодуючих пристроїв

1. Систематичний циклічний код Хеммінга

Приклад. Вихідні дані для побудови кодера систематичного циклічного коду: кількість помилок, що виправляються, s=1, отже, d_{min}= 3; кількість інформаційних символів k = 4.

По відомому k обчислюємо кількість перевірочних символів:

$$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}] = 3.$$

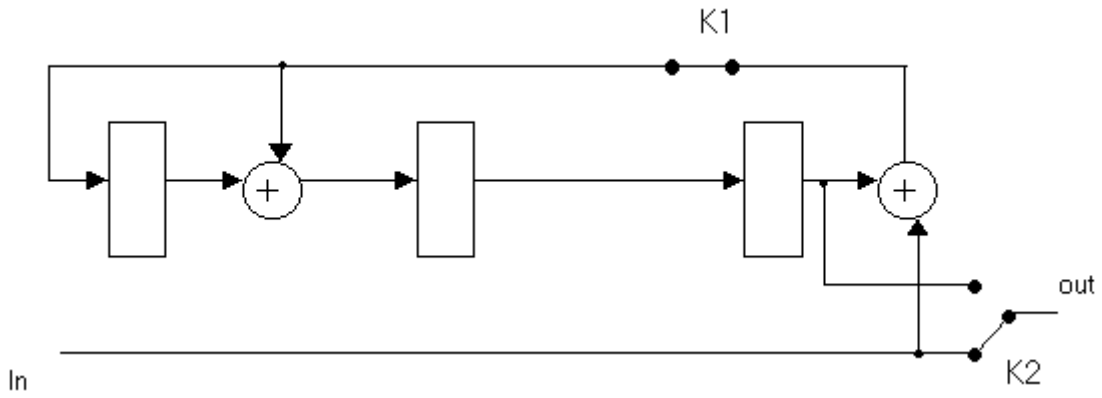
Довжина коду n = k + p = 7.

Степінь породжувального полінома дорівнює кількості перевірочних символів:

$$\deg K(x) = p = 3.$$

З таблиці незвідних поліномів вибираємо породжувальний поліном третього степеня:

$$K(x) = x^3 + x + 1.$$

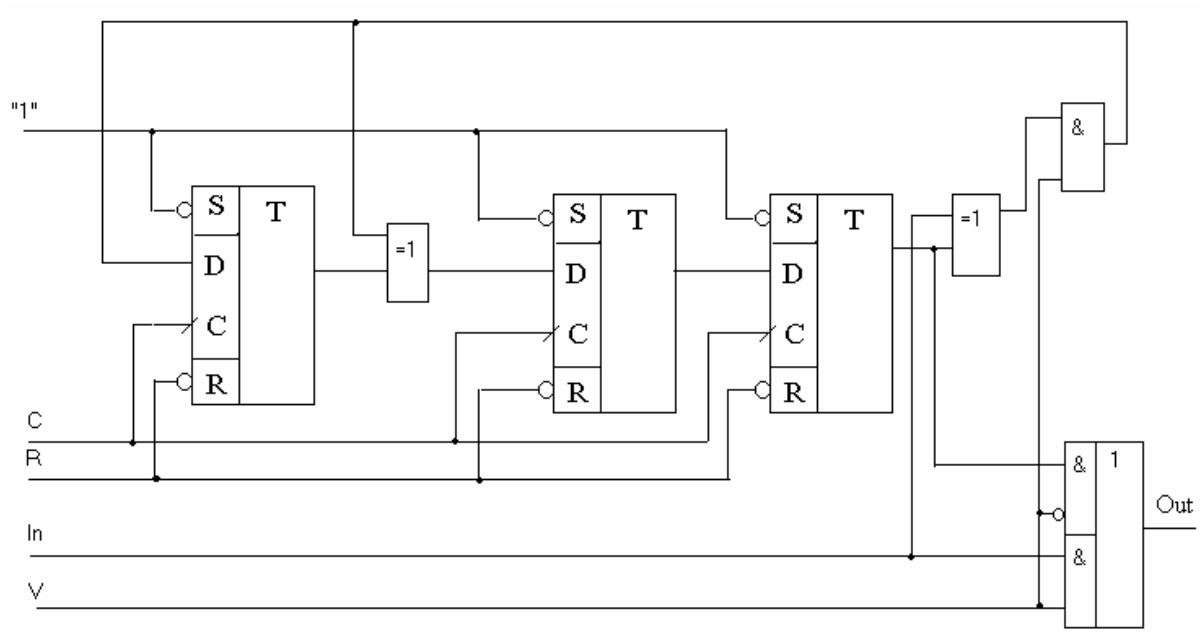


Кодуючий пристрій систематичного циклічного коду Хеммінга являє собою схему множення інформаційного полінома на поліном X^P і ділення отриманого добутку на породжувальний поліном. Робота ключів К1 і К2 схеми:

k = 4 такти : К1 замкнутий, К2 униз;

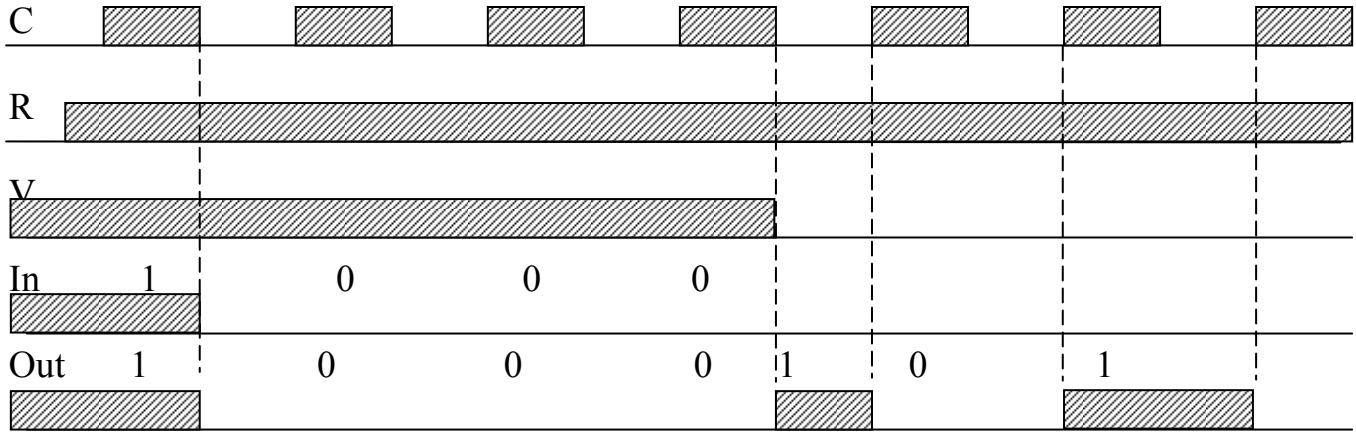
p = 3 такти : К1 розімкнутий, К2 догори.

Принципова схема кодера систематичного коду Хеммінга:



Перші k тактів інформаційна послідовність відразу передається в канал. Разом з тим вона збільшується на X^P і ділиться на породжувальний поліном. Через k тактів уміст регістра зсуву з лінійним зворотним зв'язком являє собою перевірочні символи, ключ К1 розмикається, регістр зсуву з лінійним зворотним зв'язком перетворюється в звичайний регістр, ключ К2 нагору, і протягом наступних p тактів перевірочні символи передаються слідом за інформаційними.

Часові діаграми роботи схеми (без імітації помилок):



Для часових діаграм рекомендується:

- період синхроімпульсів С брати рівним 100 нс і шпаруватість рівну 2;
- активний сигнал скидання R – 25 нс;
- керуючий сигнал V і інформаційний сигнал In змінювати під час неактивного фронту С (активний у даному випадку – передній фронт, оскільки по цьому фронту спрацьовують елементи пам'яті - тригери ТМ2);
- інформаційні символи вибирати рівними 1 і всі інші (k-1) – нулі (у цьому випадку кодове слово – перший рядок породжувальної матриці);
- кодове слово перевіряти на виході **буферного регістру декодера** (на цьому виході всі символи кодового слова мають однакову тривалість 100 нс – за умови, що С = 100 нс).

2. Несистематичний циклічний код Хеммінга

Приклад. Вихідні дані для побудови кодера систематичного циклічного коду: кількість помилок, що виправляються, s=1, отже, d_{min}= 3; кількість інформаційних символів k = 4.

По відомому k обчислюємо кількість перевірочних символів:

$$p = [\log_2\{(k + 1) + [\log_2 (k + 1)]\}] = 3.$$

Довжина коду n = k + p = 7.

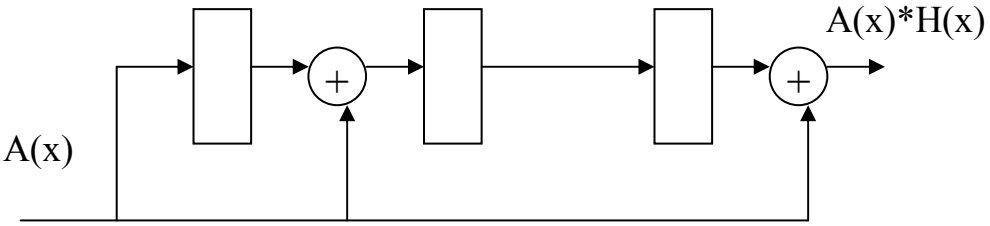
Степінь породжувального полінома дорівнює кількості перевірочних символів:

$$\deg K(x) = p = 3.$$

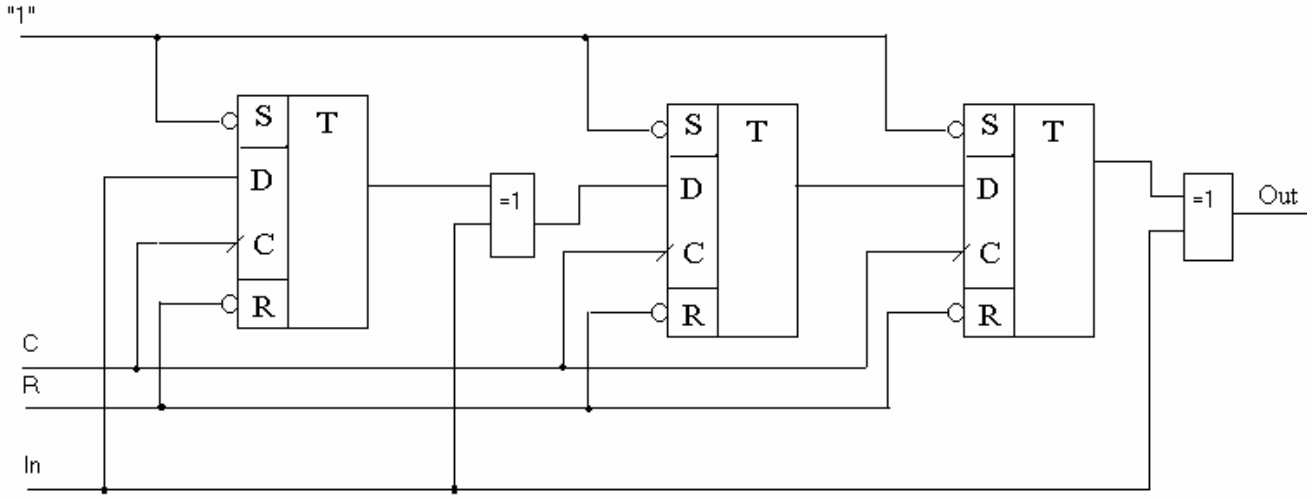
З таблиці незвідних поліномів вибираємо породжувальний поліном третього степеня:

$$K(x) = x^3 + x + 1.$$

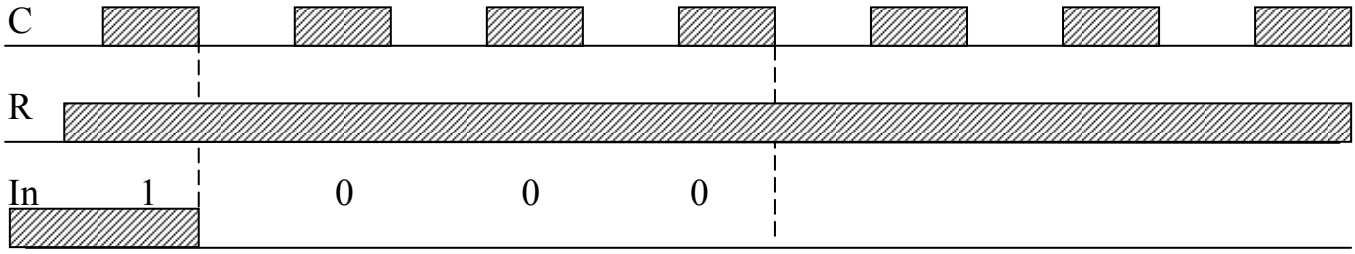
Кодуючий пристрій несистематичного циклічного коду Хеммінга являє собою схему множення інформаційного полінома на породжувальний поліном.



Принципова схема кодера несистематичного коду Хеммінга:



Часові діаграми роботи схеми (без імітації помилок):

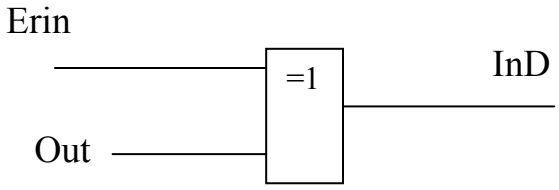


Для часових діаграм рекомендується:

- період синхроімпульсів C брати рівним 100 нс і шпаруватість рівну 2;
- активний сигнал скидання R – 25 нс;
- інформаційні символи вибирати рівними 1 і всі інші (k-1) – нулі (у цьому випадку кодове слово – перший рядок породжувальної матриці);
- кодове слово перевіряти на виході **буферного регістру декодера** (на цьому виході всі символи кодового слова мають однакову тривалість 100 нс – за умови, що C = 100 нс).

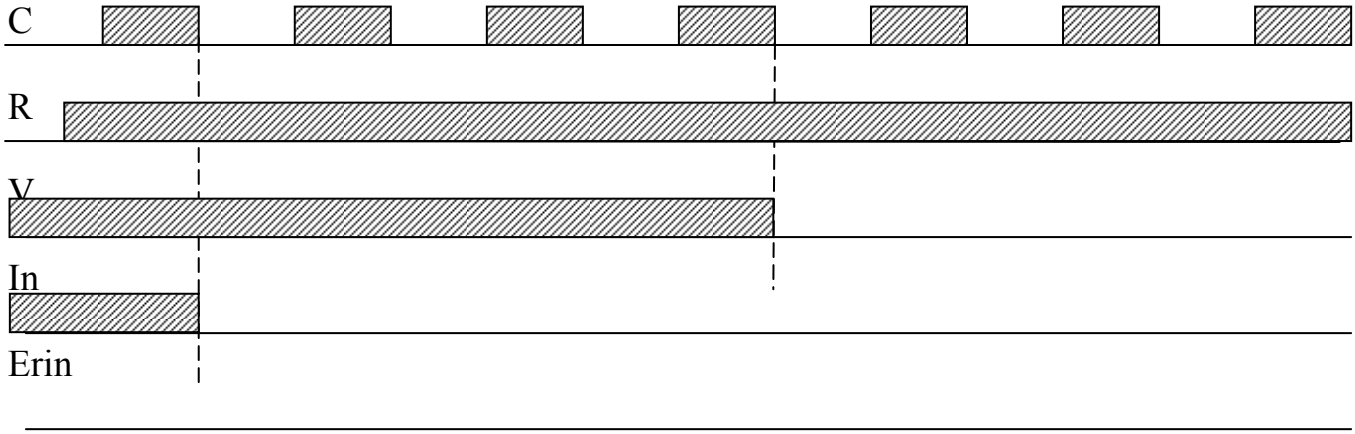
Імітація помилок.

Імітація помилок виконується за допомогою сигналу erin:

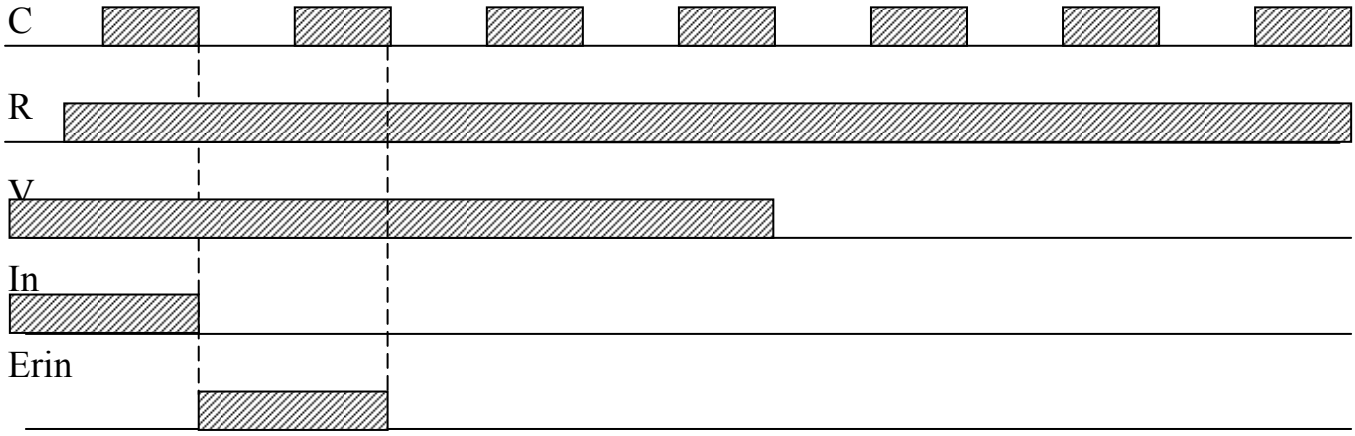


де Erin (error input) – вхід імітації помилок; Out – вихід кодера; In – вхід декодера. Необхідно одержати три часові діаграми (приклад систематичного коду):

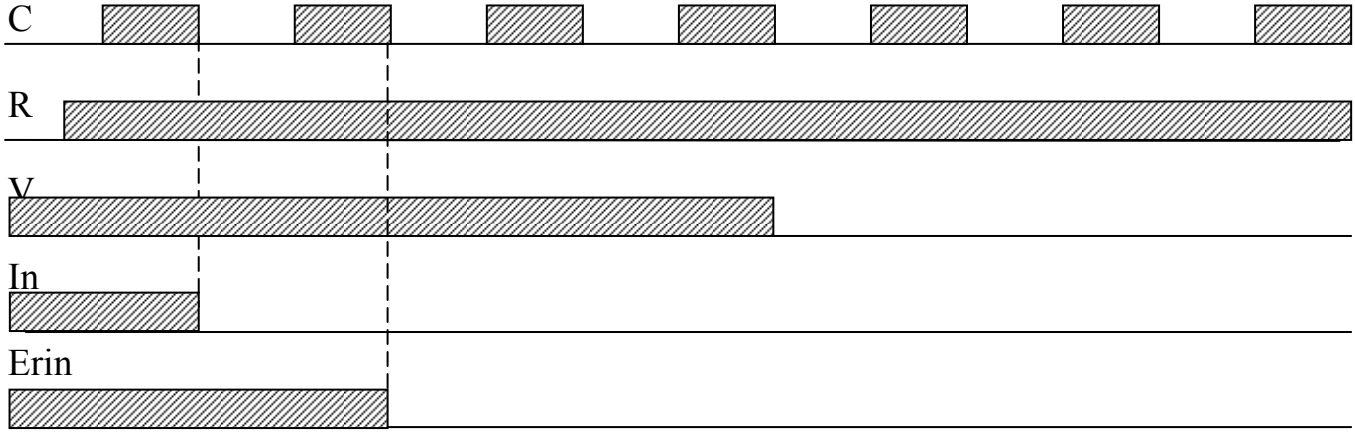
1. Без імітації помилок:



2. З імітацією одиночної помилки:



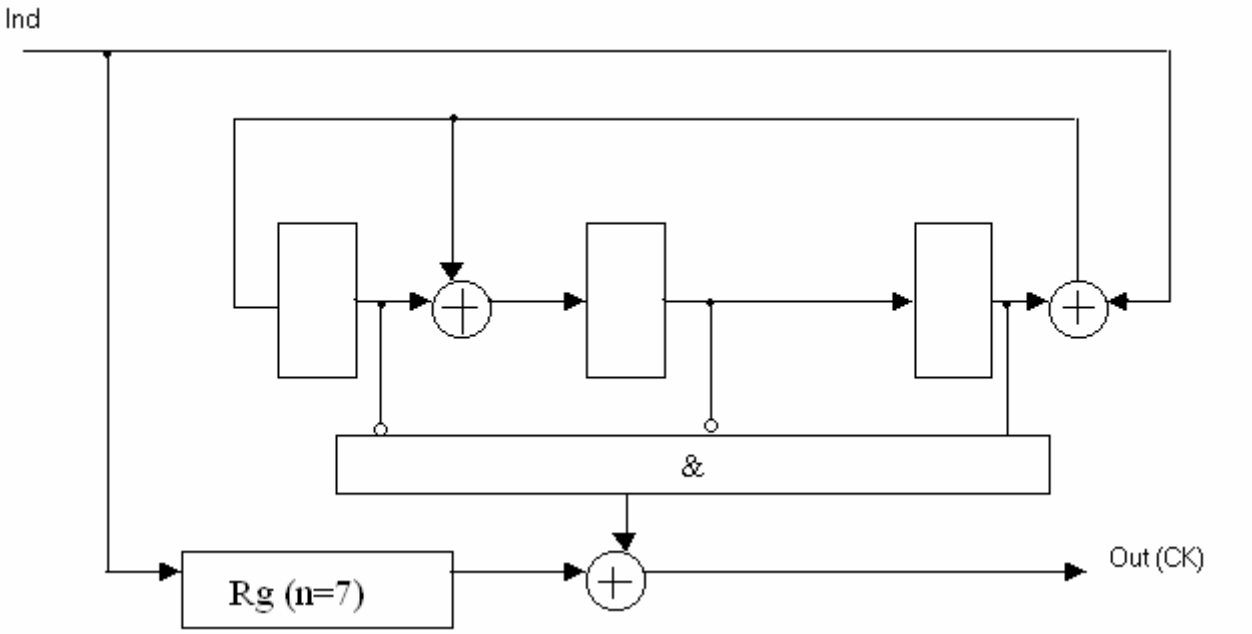
3. З імітацією подвійної помилки:



Декодер Меггітта

При розгляді принципу побудови циклічних кодів Хеммінга були сформульовані основні операції, що повинен виконувати декодер. Для формування синдрому він повинен ділити прийняте слово на породжувальний поліном.

Декодер Меггітта виконує не тільки операцію ділення на породжувальний поліном K(x), але також, як і кодер циклічних кодів, операцію множення на поліном X^P.



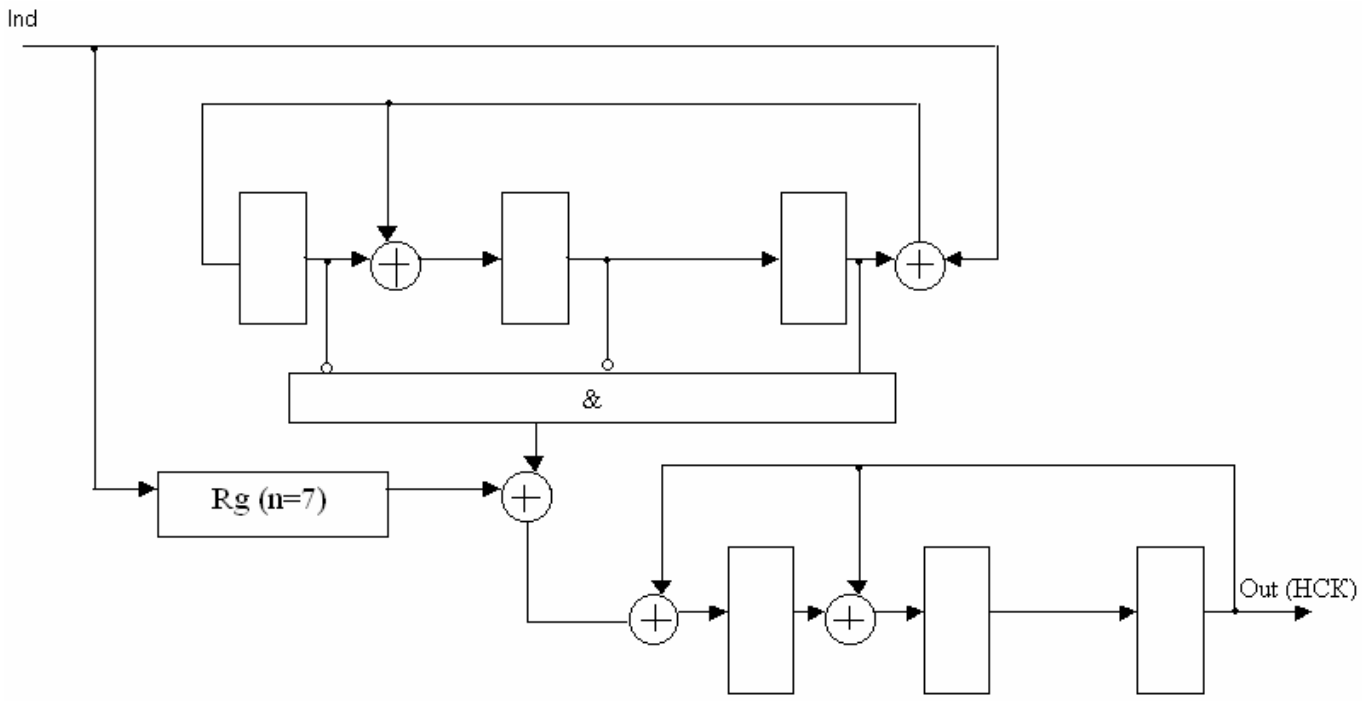
Функціональна схема декодера Меггітта для систематичного (7, 4)-коду Хеммінга з породжувальним поліномом $K(x) = x^3 + x + 1$.

Декодер працює 2n тактів.

Перші n тактів роботи: прийняте слово записується в буферний регістр (Rg) і разом з тим збільшується на X^P і ділиться на K(x). Через n тактів регістр зсуву з лінійним зворотним зв'язком (генератор синдрому) містить синдром. Якщо помилок не було, він дорівнює нулеві. Якщо в прийнятому кодовому слові мала місце одиночна помилка, синдром відмінний від нуля.

Другі n тактів: у випадку систематичного циклічного коду і відсутності помилок кодове слово з'являється на виході декодера без змін.

У випадку несистематичного коду прийняте слово ділиться на поліном, що породжує, для відновлення інформаційних символів.



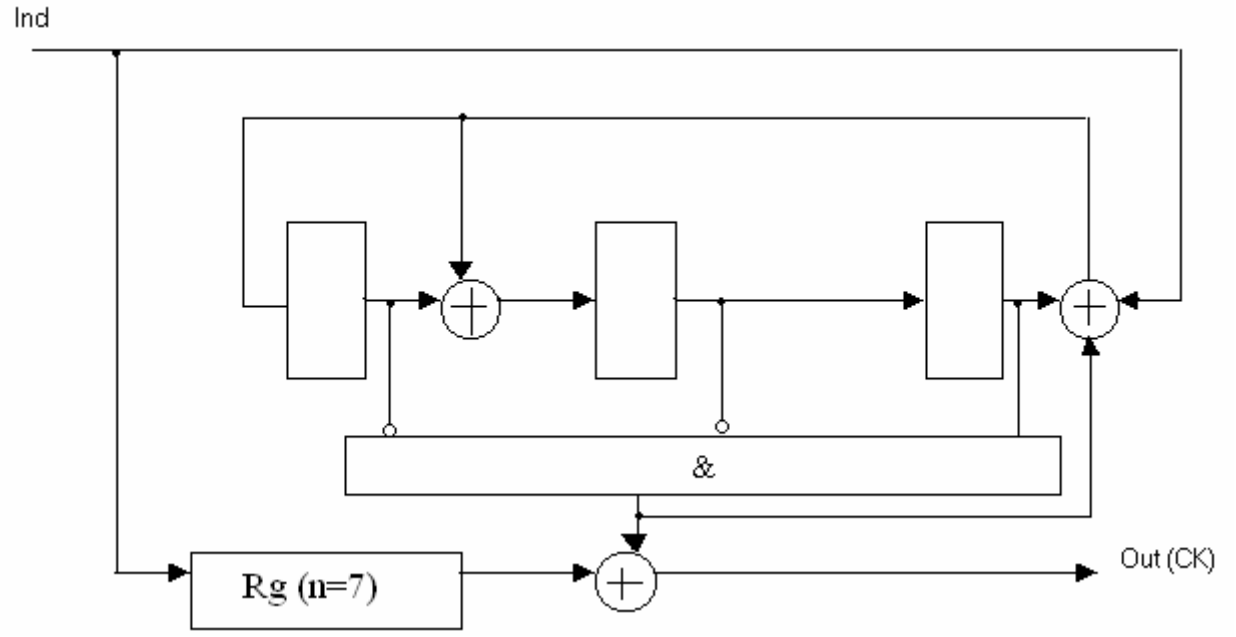
Функціональна схема декодера Меггітта для несистематичного (7, 4)-коду Хеммінга з породжувальним поліномом $K(x) = x^3 + x + 1$.

У випадку ненульового синдрому генератор синдрому продовжує свою роботу, і в момент появи в ньому комбінації „001” (у загальному випадку – комбінації з усіх нулів і останньої одиниці) на виході буферного регістру з'являється перекручений символ, що за допомогою схеми «І» та «СУМА ЗА МОДУЛЕМ ДВА» виправляється.

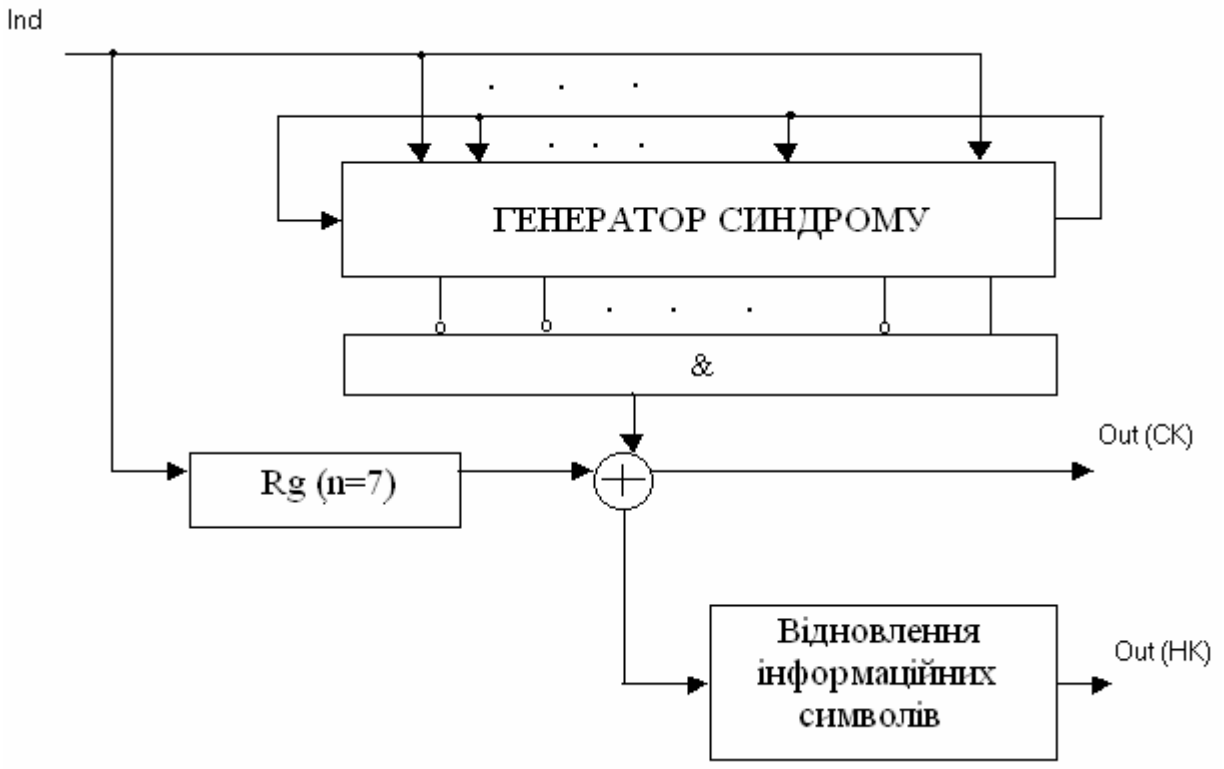
Для систематичного коду на виході з'являється виправлене кодове слово (інформаційні і перевірочні символи) через 2n тактів. Для несистематичного коду виправлене слово ділиться на породжувальний поліном для відновлення інформаційних символів, що з'являються на виході декодера через 2n + p тактів.

Для кодів Хеммінга може застосовуватися модифікація синдрому, що обнуляє уміст генератора синдрому.

Модифікація синдрому повинна виконуватися в другі n тактів. Модифікація синдрому не повинна заважати правильному формуванню синдрому в перші n тактів роботи. Тому при розробці принципової схеми на цей час модифікацію синдрому варто заборонити.



Функціональна схема декодера Меггітта для систематичного (7, 4)-коду Хеммінга з породжувальним поліномом $K(x) = x^3 + x + 1$ з модифікацією синдрому



Узагальнена схема декодера Меггітта

Недоліком декодера Меггітта є непогодженість швидкості роботи кодера і декодера. Кодер повинен чекати n тактів, поки декодер виправить помилку. Для узгодження швидкостей роботи кодера і декодера використовується конвеєрний варіант реалізації декодера, що буде розглянутий на прикладі декодера кодів БЧХ.

Укорочені циклічні коди Хеммінга

Параметри циклічних кодів Хеммінга максимальної довжини:

p	(n,k)
2	(3,1)
3	(7,4)
4	(15,11)
5	(31,26)
6	(63,57)
.	.
.	.
.	.
p	$(2^p-1, 2^p-1-p)$

За допомогою укорочування інформаційних символів можна одержати (n-i, k-i) укорочені циклічні коди, де i – параметр укорочування, причому $0 < i < k$.

Кодуючі пристрої для укорочених і нескорочених кодів однакові.

Декодер нескороченого циклічного коду для формування синдрому виконує операцію множення прийнятого кодового слова на поліном x^p і ділення отриманого добутку на породжувальний поліном К(х).

Декодер укороченого коду для формування синдрому виконує операцію множення прийнятого кодового слова на поліном, який дорівнює залишкові від ділення полінома x^{p+i} на породжувальний поліном К(х), і ділення на породжувальний поліном К(х).

Формування синдрому	
для коду Хеммінга максимальної довжини $(2^p-1, 2^p-1-p)$	для укороченого коду Хеммінга $(2^p-1-i, 2^p-1-p-i)$, $0 < i < k$
$* x^p$	$* R_{K(x)}[x^{p+i}]$
/ K(x)	/ K(x)

Приклад. Розробити декодер для циклічного систематичного коду з заданими параметрами: кількість помилок, що виправляються, s=1, таким чином, $d_{min}=3$; довжина коду n=10.

За формулою по відомій довжині коду n визначаємо кількість перевірочних символів p:

$$p = [\log_2(n+1)] = 4.$$

Тоді кількість інформаційних символів k=n-p=6. Таким чином, розроблювальний код Хеммінга має параметри (10, 6), степінь породжувального полінома дорівнює кількості перевірочних символів:

$$\deg K(x) = p = 4.$$

З таблиці незвідних поліномів вибираємо породжувальний поліном:

$$K(x) = x^4 + x + 1.$$

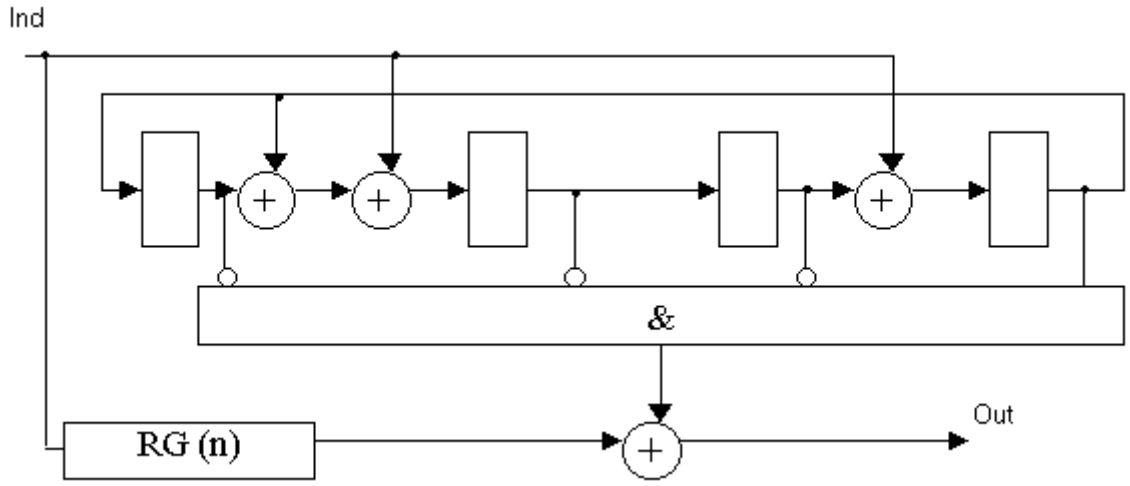
Визначимо, код укорочений чи ні. Код Хеммінга максимальної довжини для p=4 має параметри (2^p-1, 2^p-1-p) або (15, 11). Виходить, розроблювальний код (10, 6) або (15-5, 11-5) - укорочений, параметр укорочування $0 < i = 5 < k = 11$.

Кодери кодів (15, 11) і (10, 6) – однакові.

Для розробки декодера необхідно обчислити $R_{K(x)}[x^{p+i}]$:

$$
\begin{array}{l|l}
x^9 & x^4 + x + 1 \\
\hline
-x^9 + x^6 + x^5 & \\
\hline
x^6 + x^5 & x^5 + x^2 + x \\
-x^6 + x^3 + x^2 & \\
\hline
x^5 + x^3 + x^2 & \\
-x^5 + x^2 + x & \\
\hline
x^3 + x &
\end{array}
$$

$R_{K(x)}[x^{p+i}] = 1010$



Функціональна схема декодера укороченого систематичного циклічного коду Хеммінга (10, 6).

Декодер укороченого несистематичного циклічного коду Хеммінга (10, 6), на додаток до приведеної схеми, повинен ще ділити виправлене кодове слово на породжувальний поліном K(x) для відновлення інформаційних символів (так само, як і в декодері несистематичного циклічного коду Хеммінга максимальної довжини).

Двоїсті (зворотні) поліноми

Поліном

$$\mathbf{K^*(x) = x^{\deg K(x)} * K(x^{-1})}$$

називається **двоїстим (зворотним)** поліномом стосовно полінома K(x).

Приклад. Нехай заданий поліном $К(х) = х^5 + х^2 + 1$, тоді двоїстим поліномом стосовно заданого буде поліном

$К^*(х) = х^5 * (х^{-5} + х^{-2} + 1) = х^5 + х^3 + 1$.

Зворотний поліном незвідному поліному також незвідний.

Коди зворотних поліномів мають однакові характеристики, зокрема, однакові коригувальні здібності, апаратні витрати схемної реалізації кодерів і декодерів, швидкість роботи.

Коди БЧХ (Боуза – Чоудхурі - Хоквінгема)

Введення в алгебру Галуа

Групою G називається множина елементів з визначеною для кожної пари елементів операцією, що позначається *, що володіють наступними властивостями:

1. *замкнутість:* для кожної пари a і b з множини елементів c = a*b належить множині;
2. *асоціативність*: для будь-яких a, b і c з множини a*(b*c) = (a*b)*c;
3. *існування одиниці:* у множині існує елемент e, називаний одиничним, такий, що a*e = e*a = a для будь-якого елемента a множини;
4. *існування зворотних елементів*: для будь-якого a з множини існує деякий елемент b з множини, називаний зворотним a і такий, що a*b = b*a = e.

Якщо група G містить кінцеве число елементів, то вона називається кінцевою групою, а число елементів G називається порядком G.

Групи, що володіють властивістю комутативності a*b = b*a, називаються ***комутативними або абелевими*** групами.

Полем називається множина із двома визначеними на ній операціями – додаванням і множенням, причому мають місце наступні аксіоми:

1) множина утворює абелеву групу за додаванням;

2) поле замкнуте щодо множення, і множина ненульових елементів утворює абелеву групу за множенням;

3) виконується закон дистрибутивності:

$(a + b)c = ac + bc$ для будь-яких a,b і c з поля.

Одиничний елемент щодо додавання прийнято позначати через 0 і називати нулем; адитивний зворотний елементові a елемент - через –a.

Одиничний елемент щодо множення позначають через 1 і називають одиницею; мультиплікативний зворотний до елемента a елемент - через a^{-1}.

Під вирахуванням (a - b) розуміється a + (- b); під діленням (a / b) розуміється (b^{-1})*a.

Широко відомі наступні приклади полів:

1) R: множина речовинних чисел;

2) C: множина комплексних чисел;

3) Q: множина раціональних чисел;

Усі ці поля містять нескінченну множину елементів. Ми цікавимося полями з кінцевим числом елементів.

Поле з q елементами, якщо воно існує, називається кінцевим полем або ***полем Галуа*** і позначається через GF(q).

Мінімальне поле GF(2):

+	0	1
0	0	1
1	1	0

*	0	1
0	0	0
1	0	1

Нехай F - деяке поле. Підмножина в F називається ***підполем***, якщо вона саме є полем щодо наслідуваних з F операцій додавання і множення. У цьому випадку вихідне поле F називається ***розширенням поля.***

Поліномом (багаточленом) над полем GF(q) називається математичний вираз

$$f(x) = f_n*x^n + f_{n-1}*x^{n-1} + \ldots + f_1*x + f_0$$

де символ x називають невизначеною (фіктивною) змінною, коефіцієнти $f_n,...,f_0$ належать полю GF(q), а індекси і показники степенів є цілими числами.

Нульовим поліномом називається поліном f(x) = 0.

Зведеним поліномом називається поліном, старший коефіцієнт f_n якого дорівнює 1.

Степенем ненульового полінома f(x) називається індекс старшого коефіцієнта f_n; степінь полінома f(x) позначається через deg f(x). ***Незвідними поліномам*** називаються поліноми, що не можуть бути представлені у виді добутку поліномів нижчих степенів з коефіцієнтами з того ж поля.

Зведені незвідні поліноми ***називаються*** простими.

Приклад. Розглянемо поле GF(4) як розширення поля GF(2) над незвідним поліномом $p(x)=x^2 + x + 1$:

$$\{0, 1, x, x+1\}.$$

Елементи поля можуть бути представлені у виді різних позначень:

Поліноміальне	Двійкове	Десяткове	Степеневе
0	00	0	0
1	01	1	x^0
x	10	2	x^1
x +1	11	3	x^2

Використовуючи поліноміальне позначення, визначимо операції додавання і множення над елементами (тут операції виконуються за модулем два і за модулем p(x)):

+	0	1	x	x + 1	*	0	1	x	x + 1
0	0	1	x	x+1	0	0	0	0	0
1	1	0	x+1	x	1	0	1	x	x+1
x	x	x+1	0	1	x	0	x	x+1	1
x+1	x+1	x	1	0	x+1	0	x+1	1	x

Елемент β називається ***коренем полінома p(x)*** або ***коренем рівняння p(x) = 0***, якщо $p(\beta) = 0$.

Примітивним елементом поля GF(q) називається такий елемент α, що всі елементи поля, за винятком нуля, можуть бути представлені у виді степеня елемента α.

Приклад. У полі GF(5) {0, 1, 2, 3, 4} примітивним є елемент 2, тому що всі елементи даного поля, за винятком нуля, можуть бути представлені у виді степеня 2:
$2^0=1$, $2^1=2$, $2^2=4$, $2^3= 8 \bmod 5 = 3$.

Примітивним поліномом p(x) над полем GF(q) називається простий поліном над GF(q), такий, що в розширенні поля, побудованому за модулем p(x), елемент, що відповідає поліномові x, є примітивним.

Прикладом примітивного полінома є поліном $x^2 + x + 1$.

Нехай GF(q) - деяке поле, GF(Q) – розширення поля GF(q), α - елемент GF(Q). Простий поліном f(x) найменшого степеня над GF(q), для якого $f(\alpha)=0$, називається ***мінімальним поліномом*** елемента α над GF(q).

Примітивні і мінімальні поліноми визначають за допомогою таблиць.

Приклад. У таблиці задане представлення поля GF(16) як розширення поля GF(2), побудоване за примітивним поліномом $p(z) = z^4 + z + 1$.

У виді степеня	У виді полінома	У двійковому виді	Мінімальний поліном
0	0	0000	-
α^0	1	0001	x+1
α^1	z	0010	x^4+x+1
α^2	z^2	0100	x^4+x+1
α^3	z^3	1000	$x^4+x^3+x^2+x+1$
α^4	z +1	0011	x^4+x+1
α^5	$z^2 + z$	0110	x^2+x+1
α^6	$z^3 + z^2$	1100	$x^4+x^3+x^2+x+1$
α^7	$z^3 + z +1$	1011	x^4+x^3+1
α^8	$z^2 + 1$	0101	x^4+x+1
α^9	$z^3 + z$	1010	$x^4+x^3+x^2+x+1$
α^{10}	$z^2 + z +1$	0111	x^2+x+1
α^{11}	$z^3 + z^2 + z$	1110	x^4+x^3+1
α^{12}	$z^3 + z^2 + z +1$	1111	$x^4+x^3+x^2+x+1$
α^{13}	$z^3 + z^2 + 1$	1101	x^4+x^3+1
α^{14}	$Z^3 + 1$	1001	x^4+x^3+1

Можна помітити, що елементи поля $GF(2^4)$ у двійковому виді являють собою не що інше, як залишки від ділення 1 з і нулями на поліном p(z) , причому *i* відповідає

показникові степені (a^i). Таким чином, ці елементи поля можна одержати в регістрі з лінійними зворотними зв'язками (РЗЛЗЗ) з породжувальний поліномом p(z).

Якщо для елемента α відомий мінімальний поліном, то цей же поліном буде мінімальним для елемента α^2, а зворотний до цього полінома поліном буде мінімальним для елемента α^{-1}.

У таблиці незвідних поліномів для її скорочення приводять тільки ті поліноми, які не можна одержати із інших відомих поліномів у відповідності з властивостями мінімальних поліномів. Наприклад, для GF(16) побудованого за примітивним поліномом $p(z) = z^4 + z + 1$, таблиця незвідних поліномів містить тільки три мінімальних поліномів: для 1, 3, 5. Всі інші визначаються по цим поліномам, використовуючи розглянуті властивості мінімальних поліномів. Особливе положення займає мінімальний поліном для полінома α^0 – для розширень полів над поліномами будь-яких степеней він дорівнює x + 1.

Позначимо мінімальний поліном для елемента α^i як $M_i(x)$. Відповідно до визначення мінімального полінома, елемент α^i є коренем $M_i(x)$. Це значить, наприклад, якщо:

замість псевдозмінної x у поліномі $M_1(x)$ підставити z ($z=\alpha$, оскільки $p(z) = z^4 + z + 1$ – примітивний поліном), отриманий поліном за модулем p(x) дорівнює нулю: $(z^4 + z + 1) \bmod p(z) = 0$;

замість псевдозмінної x у поліномі $M_3(x)$ підставити z^3 ($z=\alpha$, оскільки $p(z) = z^4 + z + 1$ – примітивний поліном), отриманий поліном за модулем p(x) дорівнює нулю: $z^{12} + z^9 + z^6 + z^3 + 1 \bmod p(z) = 0$;

замість псевдозмінної x у поліномі $M_5(x)$ підставити z^5 (або $z^2 + z$) ($z=\alpha$, оскільки $p(z) = z^4 + z + 1$ – примітивний поліном), отриманий поліном за модулем p(x) дорівнює нулю: $(z^{10} + z^5 + 1) \bmod p(z) = 0$ і т.д.

Побудова породжувального полінома коду БЧХ

Породжувальний поліном коду БЧХ, що виправляє s помилок, повинен містити 2s коренів. Множина цих коренів:

$\{\alpha^{j0}, \alpha^{j0+1}, \alpha^{j0+2}, \ldots, \alpha^{j0+2s-1}\}$,

де j0 – конструктивний параметр. Як правило, вибирають j0 = 1.

$\{\alpha, \alpha^2, \alpha^3 \ldots \alpha^{2s}\}$

Приклад. Нехай задане поле GF(16), побудоване як розширення поля GF(2) над поліномом $p(z) = z^4 + z + 1$. Позначимо мінімальні поліноми в такий спосіб:

Елемент	Мінімальний поліном
α	$M_1(x) = x^4 + x + 1$
α^3	$M_3(x) = x^4 + x^3 + x^2 + x + 1$
α^5	$M_5(x) = x^2 + x + 1$
α^7	$M_7(x) = x^4 + x^3 + 1$

1. Якщо взяти s=1, для двійкового випадку код збігається з кодом Хеммінга.

2. Нехай s=2. Породжувальний поліном коду БЧХ, що виправляє подвійні помилки:

$B(x) = НЗК\ \{M_1(x), M_2(x), M_3(x), M_4(x)\} = НЗК\ \{M_1(x), M_3(x)\} =$
$(x^4 + x + 1)*(x^4 + x^3 + x^2 + x + 1) = x^8 + x^7 + x^6 + x^4 + 1$ (тому що $M_1(x)$, $M_2(x)$, $M_4(x)$ – той самий поліном, НЗК – найменше загальне кратне).

Тому що поле GF(16) містить 15 ненульових елементів, то довжина коду n = 15: кількість перевірочних символів p = deg B(x) = 8, кількість інформаційних символів k = n-p= 7.

Одержали код БЧХ (15, 7), що виправляє подвійні помилки.

3. Нехай s=3. Породжувальний поліном коду БЧХ, що виправляє потрійні помилки:

$B(x) = НЗК\ \{M_1(x), M_3(x), M_5(x)\} = (x^4 + x + 1)*(x^4 + x^3 + x^2 + x + 1)*(x^2 + x + 1) = x^{10} + x^8 + x^5 + x^4 + x^2 + x + 1.$

Довжина коду n = 15, тому що розглянуте поле GF(16) містить 15 ненульових елементів, кількість перевірочних символів p = deg B(x) = 10, кількість інформаційних символів k = n-p = 5.

Одержали код БЧХ (15,5), що виправляє потрійні помилки.

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

$B(x) = НЗК\{M_1(x), M_3(x), M_5(x), M_7(x)\} = (x^4 + x + 1)*(x^4 + x^3 + x^2 + x + 1)*(x^2 + x + +1)*(x^4 + x^3 + 1) = x^{14} + x^{13} + x^{12} + x^{11} + x^{10} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + x^2 + x+1.$

Довжина коду n=15, кількість перевірочних символів p=14, кількість інформаційних символів k=1.

Вийшов породжувальний поліном (15,1) коду БЧХ. Це простий код з повторенням, що виправляє сім помилок (при s = 5,6,7 – виходить такий же поліном).

Принцип побудови коду БЧХ

Як породжувальний поліном для коду БЧХ вибирається поліном, що являє собою добуток двох поліномів:

$$\mathbf{B(x) = M_1(x)*M_3(x)}$$

Ці поліноми вибираються таким чином, що синдром, отриманий для $M_1(x)$ S1 і синдром, отриманий для $M_3(x)$ S3 у випадку виникнення одиночної помилки зв'язані між собою співвідношеннями:

$$S3_i = S1_i^3$$

У випадку подвійної помилки, що займає позиції *i* і *j*:

$$S1 = S1_i + S1_j$$
$$S3 = S3_i + S3_j$$
$$S3_i = S1_i^3$$
$$S3_j = S1_j^3$$

Таким чином, одержали систему рівнянь із двома невідомими:

$$S1 = S1_i + S1_j$$

$$S3 = S1_i^3 + S1_j^3$$

вирішивши яку, можна визначити номера позицій помилок і і j, а отже, виправити помилки.

Якщо розглядати потрійну помилку s=3, то одержимо систему з трьох рівнянь із трьома невідомими і т.д.

Апаратна реалізація кодів БЧХ

Кодуючі пристрої для кодів БЧХ будуються аналогічно кодуючим пристроям для циклічних кодів Хеммінга. Відмінності полягають у реалізації декодера.

Існує три варіанти реалізації декодера, які розглянемо на прикладі коду БЧХ (15, 7), що виправляє дві помилки: породжувальний поліном $B(x) = x^8 + x^7 + x^6 + x^4 + 1$, довжина коду n=15, кількість перевірочних символів $p = \deg B(x) = 8$, кількість інформаційних символів $k = 7$.

Варіант 1.

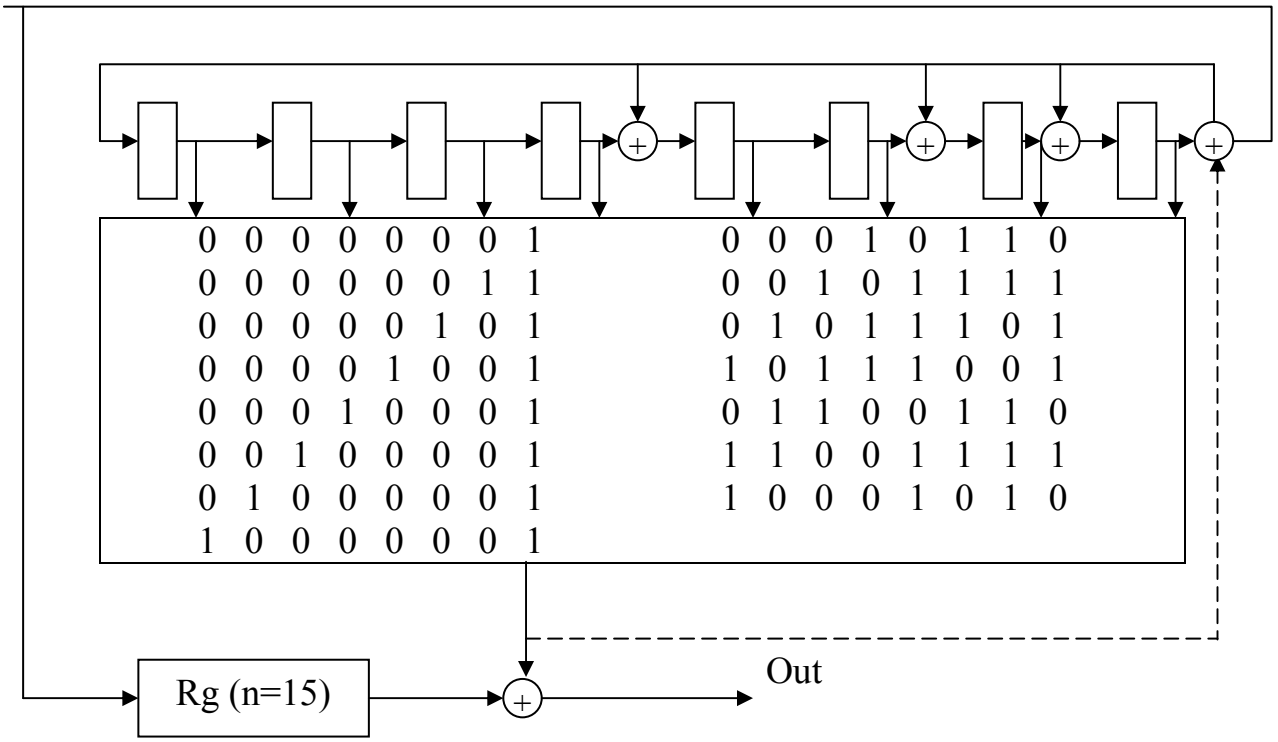
Для аналізу коду досить розглядати не реальні кодові слова з помилками, а так називані поліноми помилок, що у двійковому виді являють собою комбінацію нулів і одиниць і містять 0 на позиції, де немає помилок і 1 на позиції, де помилка має місце. Таким чином, поліноми помилок відображають конфігурацію помилок, кожної з яких можна поставити у відповідність синдром. Досить розглядати не всі конфігурації помилок, зважаючи на властивість циклічності.

Якщо n – довжина кодового слова, то виконується перевірка n конфігурацій помилок, кожна з яких містить одиницю в першій позиції. Одна з них відповідає одиночній помилці, інші – подвійним.

n=15 1 0000 00000 00000 x^{14}
1 1000 00000 00000 $x^{14} + x^{13}$
1 0100 00000 00000 $x^{14} + x^{12}$
…….....
1 0000 00000 00001 $x^{14} + 1$

Для кожної з цих 15 конфігурацій помилок обчислюється синдром, складається таблиця.

1. $S(x) = R_{B(x)}[x^{14} * x^8] = x^7$;
2. $S(x) = R_{B(x)}[(x^{14} + x^{13}) * x^8] = x^7 + x^6$;
3. $S(x) = R_{B(x)}[(x^{14} + x^{12}) * x^8] = x^7 + x^5$ і т.д.



Декодер працює 2n тактів. Перші n тактів формується синдром, другі n тактів виконується виправлення спочатку однієї, а потім другої помилки.

Варіант 2.

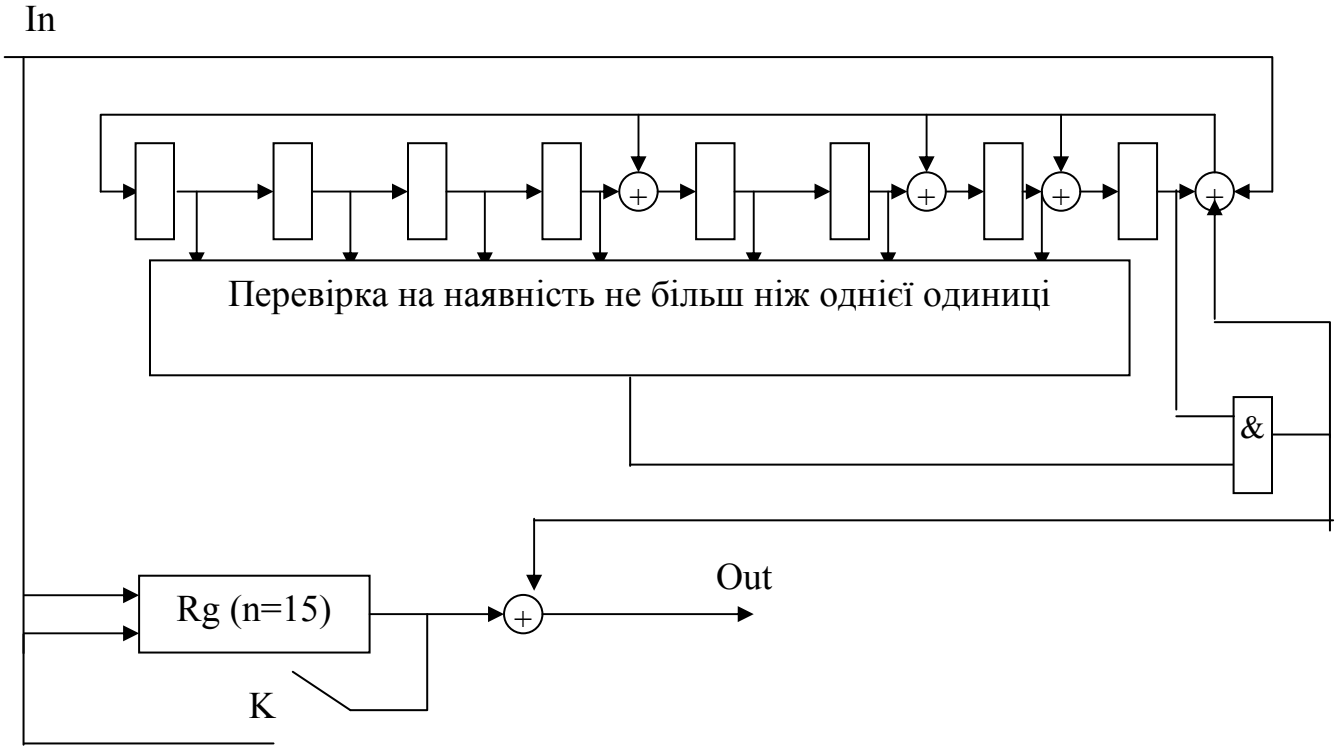
Ліва частина таблиці нерегулярна. Для спрощення схеми використовуємо властивість синдрому. Розглянемо дві конфігурації помилок:

100**1**0 00000 00000

Позначимо другу одиницю і виконаємо циклічний зсув так, щоб ця одиниця виявилася на першому місці:

10000 00000 00100

Якщо синдром для першої конфігурації помилок попадає в ліву частину таблиці, то синдром для другої конфігурації – у праву, і навпаки. Тому нерегулярну частину видаляють для спрощення схеми декодера.

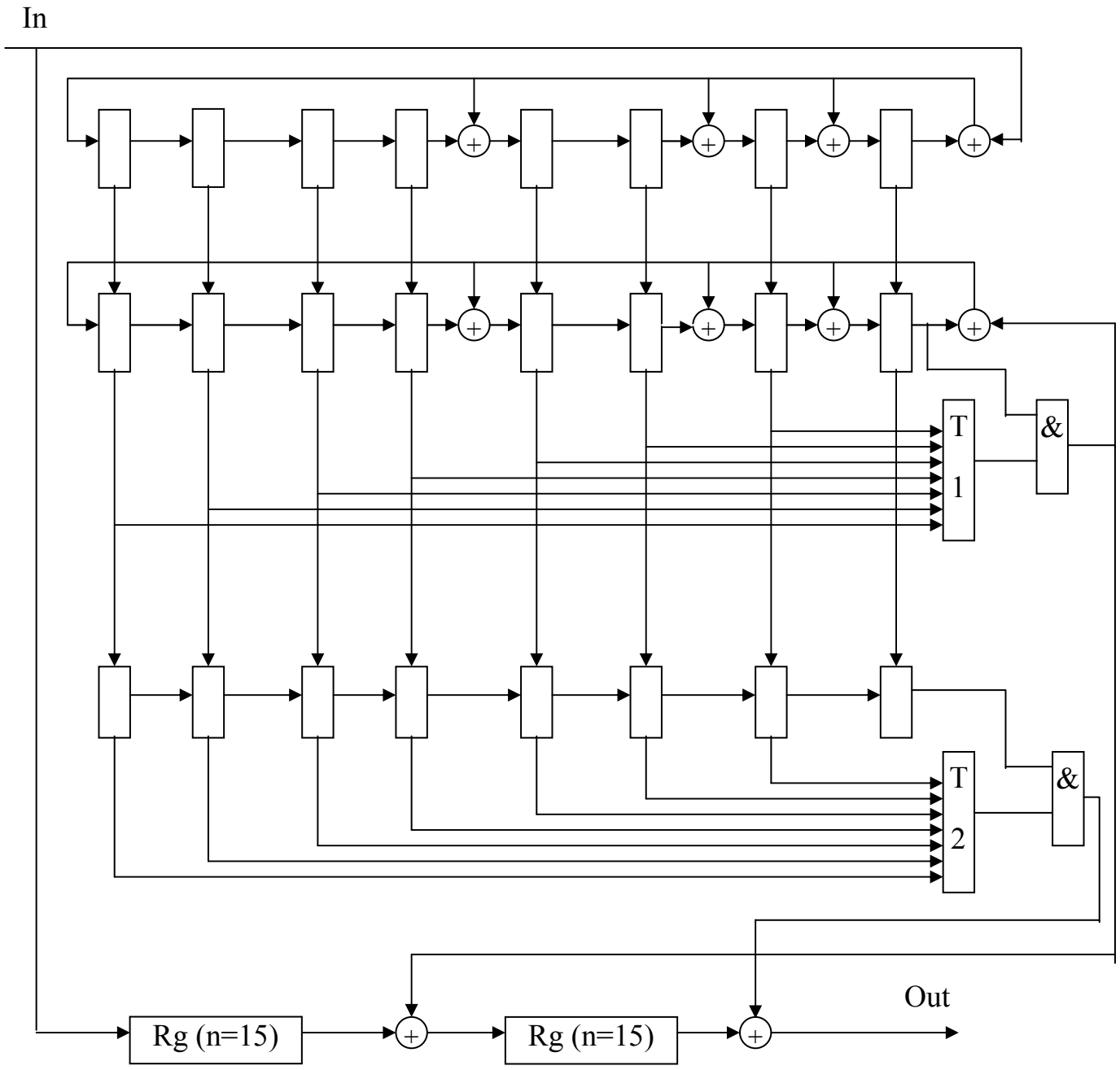


Декодер працює 3n тактів. Перші n тактів: формується синдром, кодове слово заноситься в буферний регістр. Другі n тактів: виправляється одна з існуючих помилок і кодове слово знову записується в буферний регістр (ключ К – замкнутий), треті n тактів: виправляється друга помилка. Модифікація синдрому для даного коду обов'язкова.

Недоліком даної схемної реалізації є неузгодженість роботи кодера і декодера. Кодер повинен чекати 2n тактів, поки декодер виконає виправлення.

Для усунення цього недоліку використовується ***конвеєрний варіант*** реалізації декодера.

Варіант 3 (*конвеєрний варіант* реалізації декодера).
Для даного декодера використовуються три генератори синдрому:
1. перший формує синдром;
2. другий виправляє одну з двох можливих помилок;
3. третій виправляє другу помилку.



T1 – перевірка на наявність не більш однієї одиниці;
T2 – перевірка на всі нулі.

У цьому випадку робота кодера і декодера узгоджена.

Циклічні коди, що виправляють пакети помилок

Циклічним пакетом довжини b називається вектор, усі ненульові компоненти якого розташовані серед b послідовних (по циклу) компонентів, перший й останній з яких відмінні від нуля.

0 0 | 1 0 1 1 | 0 0 0 – b=4

1 | 0 0 0 0 0 | 1 0 0 – b=4

Будь-який блоковий код, що виправляє пакети помилок довжини b, повинен містити, принаймні, 2b перевірочних символів (границя Рейгера).

Для кодів, що виправляють пакети помилок, z=p-2b (де p – кількість перевірочних символів, b – довжина пакета помилок, що виправляється) називається параметром міри неефективності.

Циклічні коди, що виправляють пакети помилок:

- коди, знайдені за допомогою ЕОМ;
- коди, знайдені аналітичним способом:
  1) коди Файра;
  2) коди Бартона;
  3) коди Ріда – Соломона, що виправляють одиночну помилку;
  4) коди Ріда – Соломона, що виправляють t помилок;
  5) коди, отримані з перерахованих вище методом укорочування і/або символьного або блочного чергування.

1. Коди, знайдені за допомогою ЕОМ

Породжувальний поліном	Параметри (n, k)	Довжина пакета b
$x^4+x^3+x^2+1$	(7,3)	2
$x^5+x^4+x^2+1$	(15,10)	2
$x^6+x^5+x^4+x^3+1$	(15,9)	3
$x^6+x^5+x^4+1$	(31,25)	2
$x^7+x^6+x^5+x^3+x^2+1$	(63,56)	2
$x^8+x^7+x^6+x^3+1$	(63,55)	3
$x^{12}+x^8+x^5+x^3+1$	(511,499)	4
$x^{13}+x^{10}+x^7+x^6+x^5+x^4+x^2+1$	(1023,1010)	4

З перерахованих кодів *методом символьного чергування* степеня j можна одержати більш довгі коди, що виправляють пакети помилок довжини (j*b). При цьому,

якщо вихідний код має параметри (n, k) і породжувальний поліном g(x), то новими параметрами будуть (jn, jk) і $g(x^j)$ відповідно.

При символьному чергуванні параметр міри неефективності z збільшується в j раз, і якщо z дорівнює нулеві для вихідного коду, то для нового коду z також буде дорівнювати нулеві.

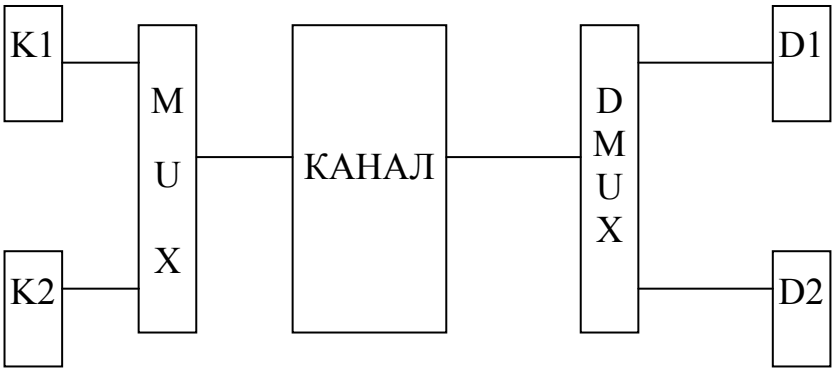
Приклад. Нехай вихідний код має породжувальний поліном $g(x) = x^4+x^3+x^2+1$ і параметри (7, 3), b=2, z = 0, то при символьному чергуванні одержуємо коди з наступними параметрами:

для j=2: $g(x) = x^8+x^6+x^4+1$, (14, 6), b=4, z=0;

для j=3: $g(x) = x^{12}+x^9+x^6+1$, (21, 9), b=6, z=0.

Принцип символьного чергування можна ілюструвати в такий спосіб. При символьному чергуванні з параметром, наприклад, j=2, можна взяти дві копії кодера, декодера, що формують і обробляють символи кодового слова по черзі.

Наприклад, якщо код (7, 3) виправляє пакет помилок довжини b=2, то взявши копію $a_1'a_2'a_3'b_1'\ b_2'b_3'b_4'$ кодового слова вихідного коду $a_1a_2a_3b_1b_2b_3b_4$, одержимо кодове слово нового коду $a_1a_1'a_2a_2'a_3a_3'b_1b_1'b_2\,b_2'b_3b_3'b_4b_4'$.

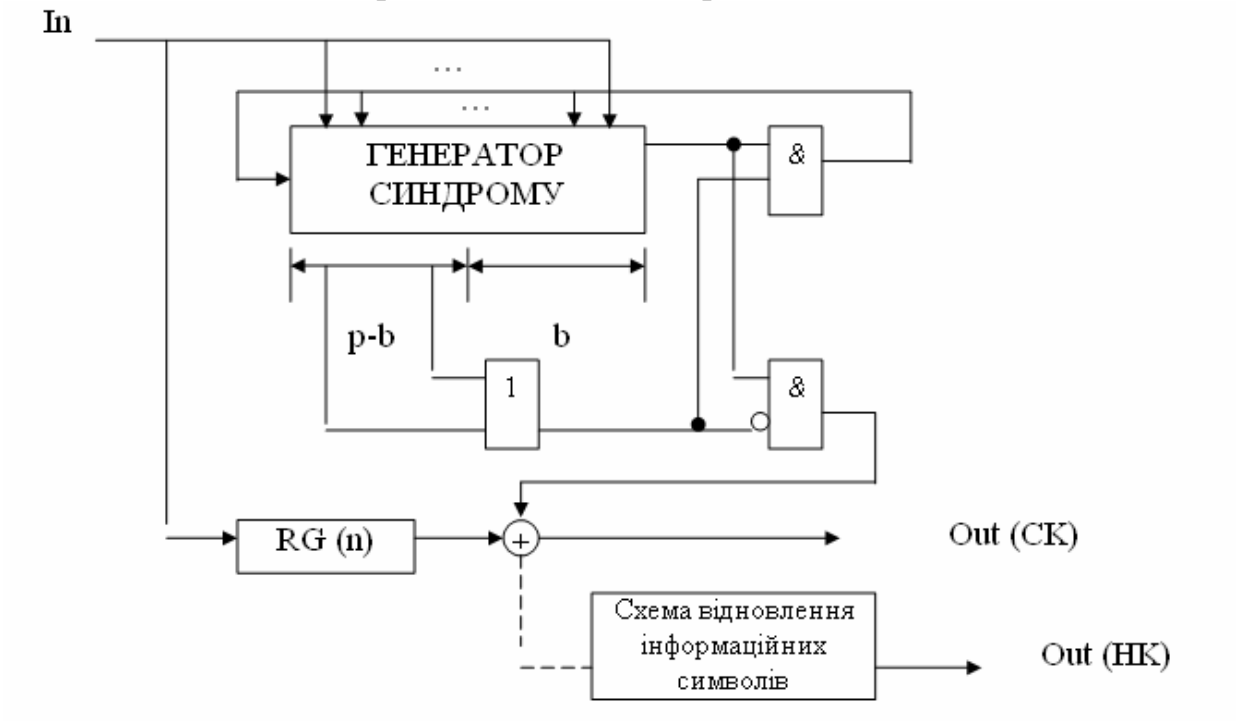


Кожна пара кодер-декодер обробляє своє кодове слово довжини 7 і може виправити пакет довжини 2. Якщо розглянути будь-який пакет помилок довжини 4 у кодовому слові $a_1a_1'a_2a_2'a_3a_3'b_1b_1'b_2\,b_2'b_3b_3'b_4b_4'$, то два помилкових поруч розташованих символи будуть відноситися до різних пар кодер-декодер, тому такий пакет буде виправлений.

Аналогічні міркування справедливі для будь-якого параметра чергування j.

Кодуючі пристрої для кодів, що виправляють пакети помилок, будуються аналогічно циклічним кодам Хеммінга, тому що і параметри, і породжувальний поліном відомі. Декодер же має свої особливості.

Узагальнена схема декодера для кодів, що виправляють пакети помилок:



Декодер працює 2n тактів. У перші n тактів формується синдром, кодове слово заноситься в буферний регістр. В другі n тактів генератор продовжує роботу і як тільки в молодших (p-b) розрядах з'являється 0, в інших b розрядах з'являється вектор помилок, що визначає конфігурацію помилок у пакеті: 1 – перекручування, 0 – вірний символ.

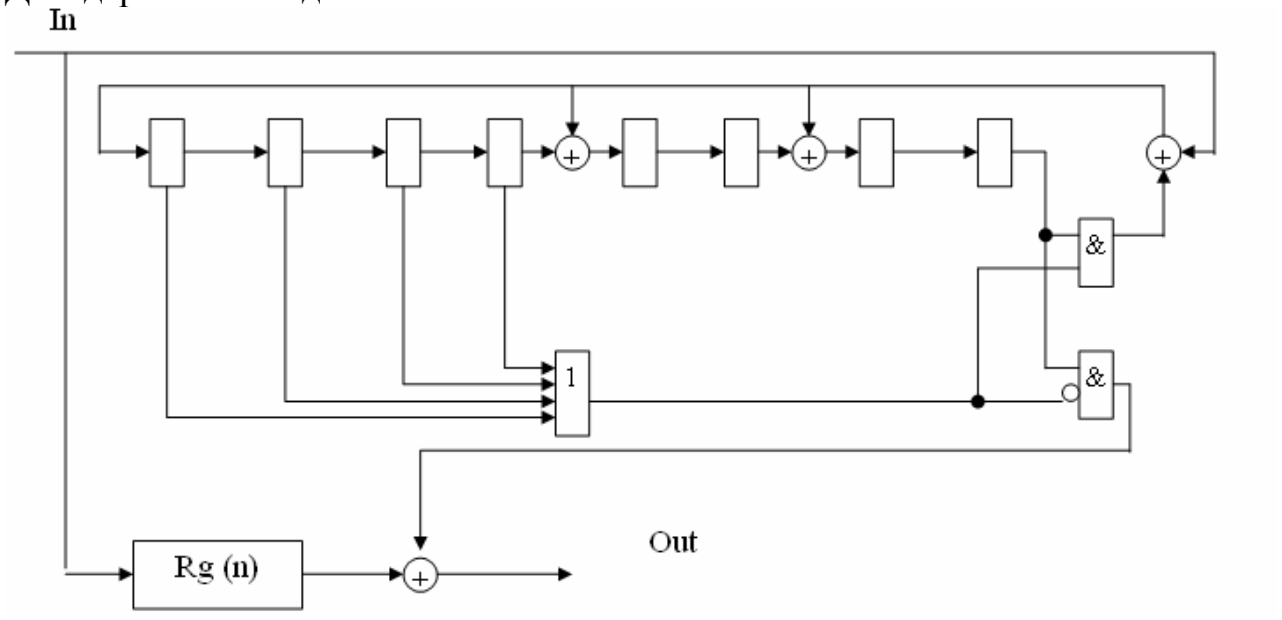
Зворотний зв'язок обривається, послідовним кодом вектор помилок складається з перекрученими символами кодового слова, виконується виправлення.

Для несистематичного коду після виправлення пакета помилок необхідне відновлення інформаційних символів діленням на породжувальний поліном.

Приклад. Вихідний код (7,3), b=2, $g(x) = x^4 + x^3 + x^2 + 1$.

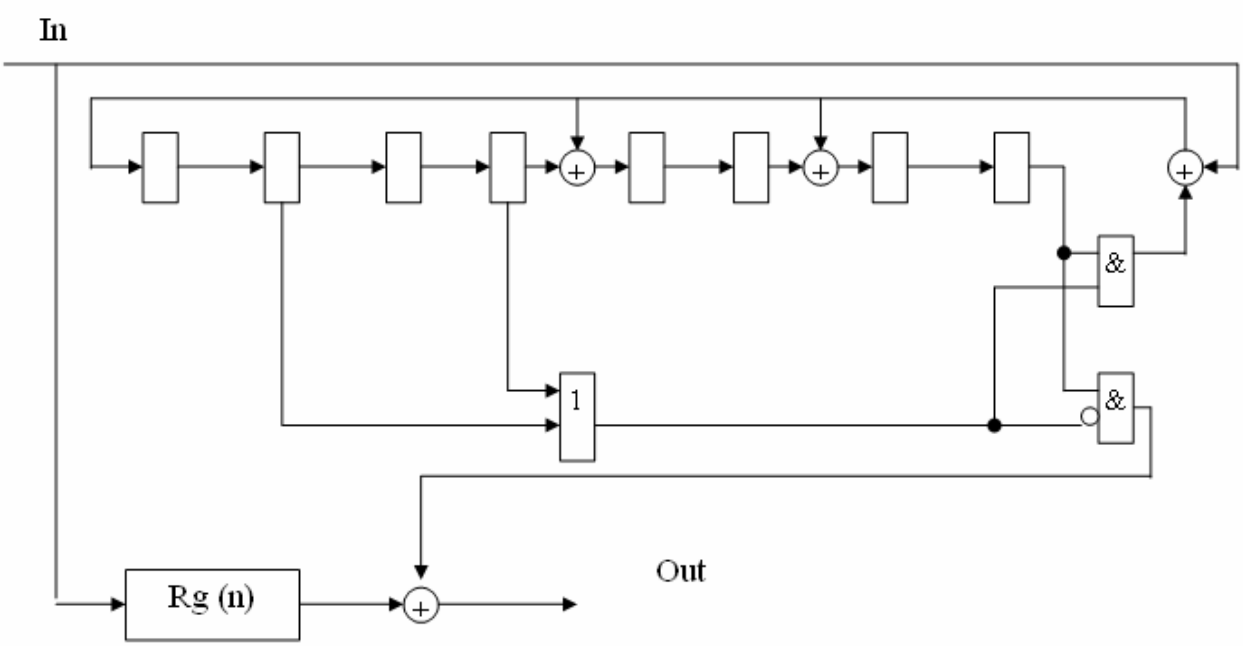
Параметр чергування j=2, початковий код (14,6), b=4, отриманий породжувальний поліном $g(x) = x^8 + x^6 + x^4 + 1$.

Декодер має вигляд:



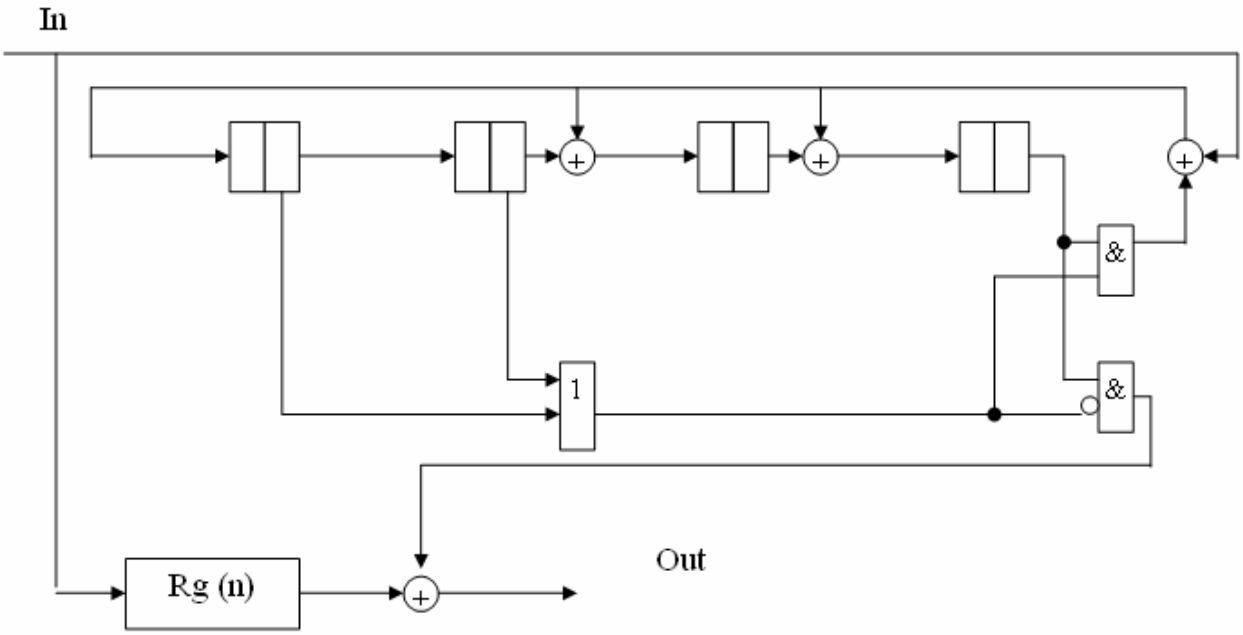
Зважаючи на ***принцип символьного чергування***, розглянутий раніше, можна помітити, що парні і непарні (при символьному чергуванні з j=2) розряди генератора синдрому ніяк не взаємодіють один з одним. Цю схему можна розглядати як два генератори синдрому в одному, котрі працюють по черзі.

Тому такий декодер можна спростити в такий спосіб:



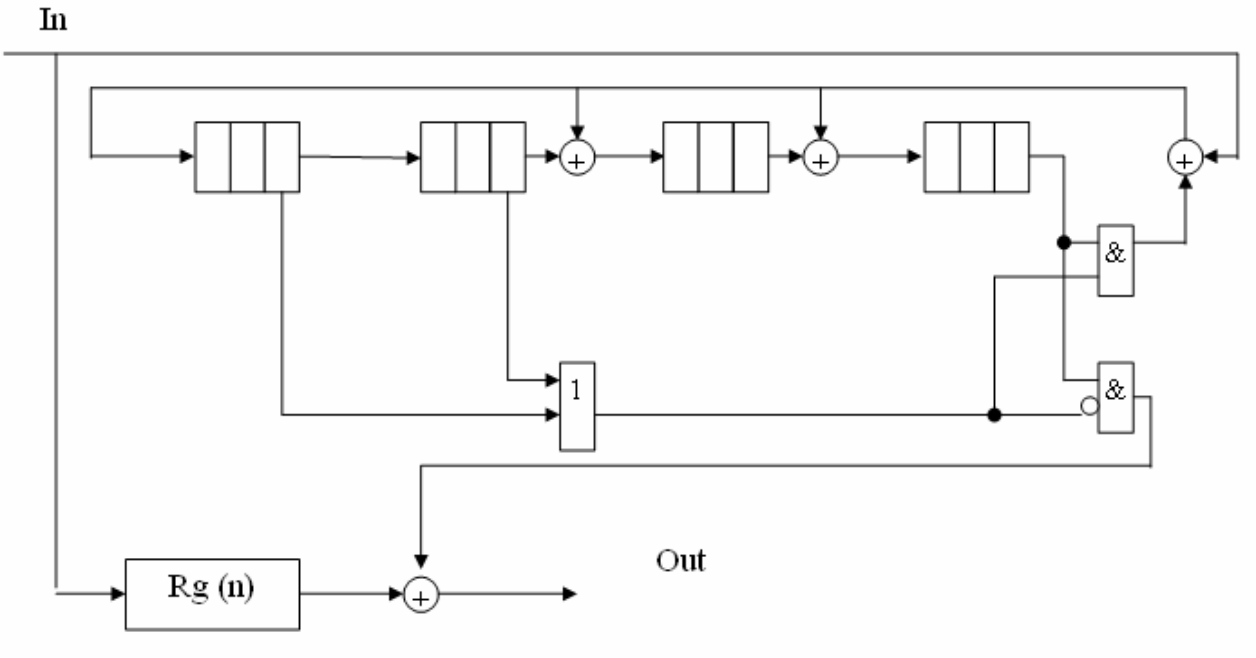
Цю схему можна розглядати як два генератори синдрому в одному, котрі працюють по черзі. При цьому вони використовують ті самі комбінаційні елементи.

Схему декодера можна зобразити в такому виді:



З цього приклада випливає, що для побудови декодера коду з параметром чергування j, у схемі декодера побудованого для вихідного коду досить замість одного використовувати j елементів пам'яті.

Наприклад, якщо параметр чергування j=3 і вихідний код має параметри (7,3), b=2, $g(x) = x^4 + x^3 + x^2 + 1$, декодер отриманого коду можна представити в наступному виді.



2. Коди Файра

Породжувальний поліном коду має вигляд

$$\mathbf{F(x) = (x^{2b-1} + 1)* P(x),}$$

де P(x) – примітивний поліном, **deg P(x) = b,** b – довжина пакета помилок, що виправляється. Довжина коду $\mathbf{n = (2^b - 1)*(2b - 1),}$ кількість перевірочних символів **p = 3b – 1,** кількість інформаційних символів **k = n – p,** параметр міри неефективності коду **z = b - 1.**

Якщо b дуже велике, то використовують укорочені коди Файра, оскільки в цьому випадку довжина коду дуже велика. Методи укорочування циклічних кодів, що виправляють пакети помилок, аналогічні методам укорочування циклічних кодів Хеммінга.

3. Коди Бартона

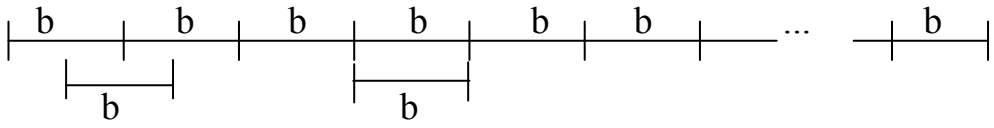
Породжувальний поліном коду Бартона має вигляд

$$\mathbf{B(x) = (x^b + 1)* P(x),}$$

де P(x) – примітивний поліном, **deg P(x) = b,** b – довжина пакета помилок, що виправляється. Довжина коду $\mathbf{n = (2^b - 1)*b,}$ кількість перевірочних символів **p = 2b,** кількість інформаційних символів **k = n – p,** параметр міри неефективності коду **z = 0.**

Для кодів Бартона існують обмеження на характер розташування пакета помилок.

Розглянемо наступну ілюстрацію цього обмеження. Кодове слово довжини n можна розбити на $2^b - 1$ ділянок по b символів.



Невиправний пакет помилок

Виправний пакет помилок

Якщо пакет помилок попадає у виділені ділянки, то він буде виправлений. Якщо ж пакет захоплює дві виділених ділянки, то такий пакет не може бути виправлений.

Щоб зняти це обмеження, використовується поблочне чергування кодів Бартона. Суть його аналогічна символьному чергуванню, тільки замість чергування символів перемежовують блоки довжини b. За допомогою додаткових копій кодів Бартона з довжиною пакета b одержують перемежовуваний код з довжиною пакета b'

$$\mathbf{b' = j*b – (b - 1).}$$

При j=2 $b' = b+1$, при j = 3 $b' = 2b + 1$ і т.д. Ефективність поблочного чергування росте зі збільшенням j.

Коди Ріда - Соломона

Коди Ріда-Соломона знайшли широке застосування в цифрових системах зв'язку і збереження інформації. Можна привести найбільш відомі приклади: (255, 223, 33) код Ріда-Соломона для космічного зв'язку NASA, укорочені коди Ріда-Соломона над полем Галуа GF(2^8) для CD-ROM, DVD і цифрового телебачення високої чіткості (формат HDTV), розширений (128, 122, 7) код Ріда-Соломона над полем Галуа GF(27) для кабельних модемів.

Породжувальний поліном кодів Ріда-Соломона

Коди Ріда – Соломона є окремим випадком кодів БЧХ. Головна відмінність кодів Ріда – Соломона полягає в тому, що для них символом виступає не двійковий символ (один біт), а елемент поля Галуа (декілька біт).

Породжувальний поліном коду Ріда - Соломона, що виправляє s помилок, повинен містити 2s коренів:

$\{\alpha^{j}_{0}, \alpha^{j_0+1}, \alpha^{j_0+2},\ldots,\alpha^{j_0+2s-1}\}$,

де j_0 – конструктивний параметр.

Як правило, j_0 вибирають рівним 1. Тоді множина коренів полінома приймає вид $\{\alpha, \alpha^2, \alpha^3\ldots\alpha^{2s}\}$...

Для коду Ріда – Соломона, що виправляє s помилок, породжувальний поліном має такий вигляд:

$$RS(x) = (x - \alpha)(x - \alpha^2)(x - \alpha^{3)}\ldots(x - \alpha^{2s}),$$

При такому представленні очевидно, що породжувальний поліном має множину коренів $\{\alpha, \alpha^2, \alpha^3\ldots\alpha^{2s}\}$...

Коди Ріда – Соломона, що виправляють одиночну помилку

Для коду Ріда-Соломона, що виправляє одиночну помилку (s=1), породжувальний поліном має вигляд $RS(x) = (x - \alpha)(x - \alpha^2)$.

Можливо кілька форм запису породжувального полінома для кодів Ріда-Соломона. Як правило, для побудови кодів Ріда–Соломона використовують розширення поля GF(2) над примітивним поліномом p(z). У цьому випадку, відповідно до визначення примітивного полінома, елемент поля z є примітивним. Тому замість позначення примітивного елемента α можна використовувати z:

$RS(x) = x^2 + (\alpha + \alpha^{2)}*x + \alpha^3 = x^2 + (z + z^2)*x + z^3$.

Інша форма буде залежати від того, яке саме поле використовується для побудови коду Ріда-Соломона. Тому цю форму розглянемо на прикладі.

Приклад. Побудуємо поле Галуа GF(8) як розширення поля GF(2) над примітивним поліномом $p(z) = z^3 + z + 1$. Елементи поля можуть бути представлені в різному позначенні.

У виді степеня	У виді полінома	У двійковому виді
0	0	000
α^0	1	001
α^1	Z	010
α^2	z^2	100
α^3	z + 1	011
α^4	z^2 +z	110
α^5	z^2 + z + 1	111
α^6	z^2 + 1	101

Оскільки $RS(x) = x^2 + (z + z^2)*x + z^3$, а $(z + z^2)$ для розглянутого поля GF(8) у степеневому позначенні α^4, z^3 - α^3, то породжувальний поліном можна представити так: $RS(x) = x^2 + \alpha^4 x + \alpha^3$.

При зміні j_0 також змінюється породжувальний поліном. Наприклад, при $j_0=2$ $RS(x) = (x - \alpha^2)(x - \alpha^3) = x^2 + (z^3 + z^2)x + z^5$.

Для GF(8) над $p(z) = z^3 + z + 1$ $RS(x) = x^2 + (z^2 + z + 1)x + z^2 + z + 1 = x^2 + \alpha^5 x + \alpha^5$.

Крім того, відзначимо, що елементи поля в загальному виді для поля GF(8) можна позначити: $a_2z^2 + a_1z + a_0$, де a_2, a_1, a_0 – коефіцієнти, що приймають різні значення. Ці елементи поля - у даному випадку тріади - є символами коду Ріда – Соломона.

Кодуючі і декодуючі схеми

Отже, для коду Ріда-Соломона, що виправляє одиночні помилки, отримані різні форми породжувальних поліномів:

1) $RS(x) = (x - \alpha)(x - \alpha^2)$ при $j_0 = 1$ – для побудови схем у такому виді не використовується;

2) $RS(x) = (x - \alpha^2)(x - \alpha^3)$ при $j_0 = 2$ - для побудови схем у такому виді не використовується;

3) $RS(x) = x^2 + (z + z^2)*x + z^3$ при $j_0 = 1$ – не залежить від конкретного поля Галуа і може бути використана для побудови схем кодера і декодера;

4) $RS(x) = x^2 + (z^3 + z^2)x + z^5$ при $j_0 = 2$ – не залежить від конкретного поля Галуа і може бути використана для побудови схем кодера і декодера;

5) $RS(x) = x^2 + (z + z^2)*x + z + 1$ при $j_0 = 1$ – залежить від конкретного поля Галуа і може бути використана для побудови схем кодера і декодера;

6) $RS(x) = x^2 + (z^2 + z + 1)x + z^2 + z + 1$ при $j_0 = 2$ – залежить від конкретного поля Галуа і може бути використана для побудови схем кодера і декодера;

7) $RS(x) = x^2 + \alpha^4 x + \alpha^3$ при $j_0 = 1$ – залежить від конкретного поля Галуа і може бути використана для побудови схем кодера і декодера;

8) $RS(x) = x^2 + \alpha^5 x + \alpha^5$ при $j_0 = 2$ – залежить від конкретного поля Галуа і може бути використана для побудови схем кодера і декодера..

Непарні і парні варіанти поліномів визначають різні коди – у них буде різна схемна реалізація, але основні параметри – коригувальні можливості, довжина коду n, кількість інформаційних k і перевірочних n-k=p символів - однакові.

Як правило, для представлення функціональних схем кодуючих і декодуючих пристроїв використовують 7, 8 варіанти породжувальних поліномів, вид схем найбільш компактний, але побудова таких поліномів потребує побудову поля Галуа.

Для розробки принципових схем краще використовувати 3-6 варіанти поліномів, оскільки вони не вимагають побудови поля Галуа, причому 5, 6 варіанти більш кращі.

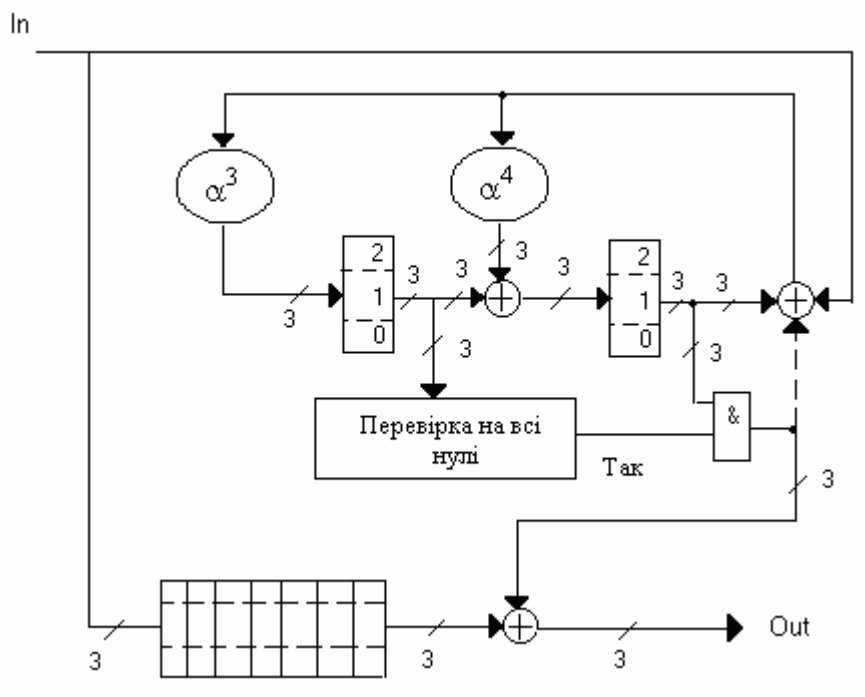
Розглянемо приклади.

Для поля GF(8) і породжувального полінома $RS(x) = x^2 + \alpha^4 x + \alpha^3$ код Ріда-Соломона, що виправляє одну помилку (одну тріаду - три двійкові символи) має наступні параметри: довжина коду n=7, кількість перевірочних символів p=degRS(x)=2, кількість інформаційних символів k=n-p=5. Таким чином, реалізуємо (7, 5)-код Ріда-Соломона, причому числа 7 і 5 означають кількість тріад двійкових символів.

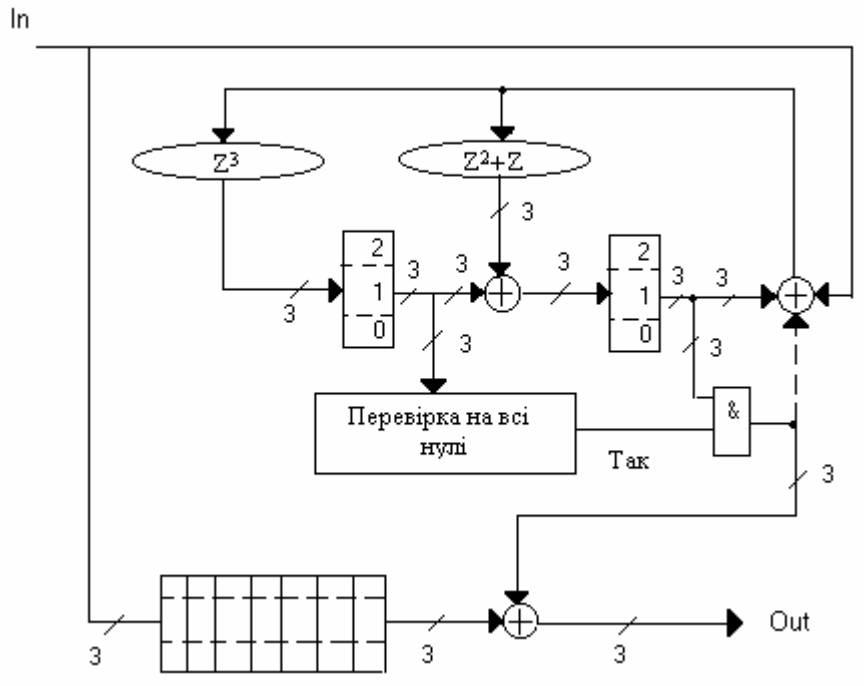
Кодер цього коду аналогічний кодеру циклічного коду Хеммінга. Як і для кодів Хеммінга, кодер систематичного коду Ріда-Соломона являє собою схему множення на поліном x і ділення на породжувальний поліном; кодер несистематичного коду Ріда-Соломона являє собою схему множення на породжувальний поліном. Різниця – різна інтерпретація символу кодового слова і, відповідно, елементів пам'яті й помножувачів на константу.

Схема декодера аналогічна декодеру Меггітта для циклічного коду Хеммінга, за винятком того, що кожний елемент затримки в даному випадку – не один елемент пам'яті, а три. Крім того, у цієї схеми специфічні помножувачі на константу. Саме ці помножувачі і представляють ускладнення при перетворенні функціональної в принципову схему. Особливості реалізації для кодерів і декодерів однакові, тому розглядати їх будемо на прикладі декодерів.

Для породжувального полінома $RS(x) = x^2 + \alpha^4 x + \alpha^3$ декодер має такий вигляд.

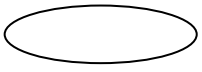


Для породжувального полінома $RS(x) = x^2 + (z + z^2)*x + z^3$ декодер має такий же вид за винятком помножувачів на константи.



Декодер для коду Ріда – Соломона (7,5)

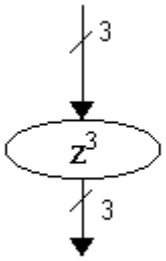
Число елементів-тріад пам'яті генератора синдрому дорівнює степені породжувального полінома RS(x). Буферний регістр складається з n елементів-тріад.

 - помножувач на константу

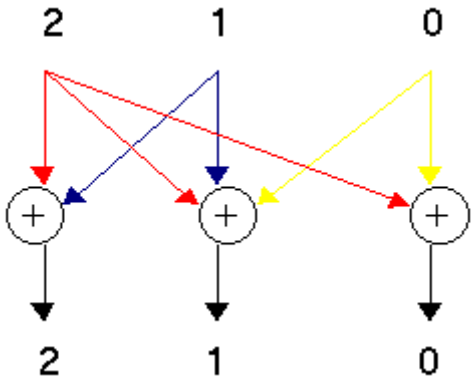
Розглянемо приклад реалізації помножувача на константу z^3.

Щоб поставити йому у відповідність схему на двійкових елементах, визначимо $(a_2z^2 + a_1z + a_0)*z^3 \mod p(z)$.

$$
\begin{array}{l|l}
a_2z^5 + a_1z^4 + a_0z^3 & z^3 + z + 1 \\
\underline{- a_2z^5 + a_2z^3 + a_2z^2} & \\
a_1z^4 + (a_2+a_0)z^3 + a_2z^2 & a_2z^2 + a_1z + (a_2+a_0) \\
\underline{- a_1z^4 + a_1z^2 + a_1z} & \\
(a_2+a_0)z^3 + (a_1+a_2)z^2 + a_1z & \\
\underline{-(a_2+a_0)z^3 + (a_2+a_0)z + (a_2+a_0)} & \\
(a_1+a_2)z^2 + (a_0+a_1+a_2)z + (a_2+a_0) &
\end{array}
$$



По отриманому залишку від ділення будується схема помножувача на константу z^3 :



Одержати залишок R від ділення для реалізації схеми помножувача на константу можна за допомогою декількох операцій множення і ділення на примітивний поліном:

1. $(a_2z^2 + a_1z + a_0)*z^2 = a_2z^4 + a_1z^3 + a_0z^2$

$$
\begin{array}{l|l}
a_2z^4 + a_1z^3 + a_0z^2 & z^3 + z + 1 \\
\underline{- a_2z^4 + a_2z^2 + a_2z} & \\
a_1z^3 + (a_2+a_0)z^2 + a_2z & a_2z + a_1 \\
\underline{- a_1z^3 + a_1z + a_1} & \\
(a_2+a_0)z^2 + (a_1+a_2)z + a_1 \quad \text{залишок R1} &
\end{array}
$$

2. $(a_2z^2 + a_1z + a_0)*z = a_2z^3 + a_1z^2 + a_0z$

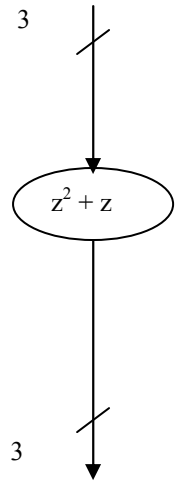
$$\begin{array}{r|l} a_2z^3 + a_1z^2 + a_0z & z^3 + z + 1 \\ \hline - a_2z^3 + a_2z + a_2 & \\ \hline a_1z^2 + (a_2+a_0)z+a_2 & a_2 \end{array}$$

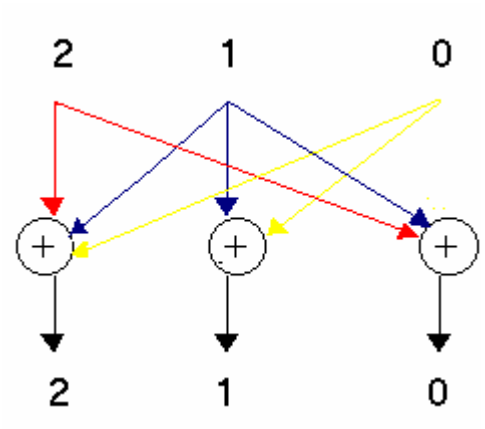
залишок R2

3. Оскільки код лінійний, R = R1 + R2

$$\begin{array}{l} (a_2 + a_0)z^2 + (a_1 + a_2)z + a_1 \\ + \quad a_1z^2 + (a_2+a_0)z+a_2 \\ \hline (a_2 + a_1 + a_0)z^2 + (a_1 + a_0)z+(a_2 + a_1) \end{array}$$

По отриманому залишку від ділення будується схема помножувача на константу $(z^2 + z)$:





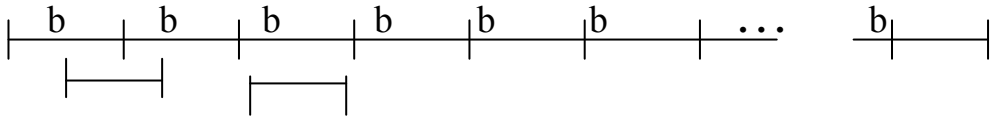
Коди Ріда-Соломона можна використовувати для виправлення помилок, як у паралельному, так і в послідовному потоці двійкових символів.

Код Ріда – Соломона (7, 5) і дозволяє виправити:

1. одну помилкову тріаду в послідовності тріад довжиною 7;
2. пакет помилок довжини 3 у послідовному коді двійкових символів з обмеженням на характер розташування пакета помилок у послідовності символів довжини 21.

Кілька зауважень із приводу зазначеного обмеження на прикладі розглянутого коду. При перетворенні тріад у послідовний код на виході кодера, послідовність двійкових символів можна розділити на ділянки по три символи. Якщо пакет помилок попадає в одну з таких ділянок, він буде виправлений. Якщо ж пакет помилок захоплює дві таких ділянки, декодер не зможе його виправити, оскільки після перетворення послідовного коду в паралельний на вході декодера, помилки будуть розташовуватися в двох тріадах. Це відповідає подвійній помилці в символах коду Ріда-Соломона, тому декодер не зможе її виправити.

У загальному випадку для коду поля Галуа $GF(2^b)$, ці ситуації можна зобразити в такий спосіб:



Невиправлюваний пакет помилок

Виправлюваний пакет помилок

Щоб зняти обмеження на характер розташування помилок, використовують символьне чергування кодів Ріда – Соломона, що виправляють одиночну помилку.

Наприклад, при символьному чергуванні коду Ріда – Соломона (7,5) з параметром чергування j = 3 одержуємо код Ріда-Соломона (21, 15), що дозволяє виправити:

- для паралельного коду пакет 3 перекручені тріади без обмеження на характер розташування помилок;
- для паралельного коду 3 перекручені тріади з обмеженням на характер розташування помилок;
- для послідовного коду пакет помилок довжини 7 (без обмеження на характер розташування помилок); пакет довжини 9 або 3 пакети довжини 3 або 1 пакет довжини 6 і 1 пакет довжини 3 (з обмеженням на характер розташування помилок).

У загальному випадку довжина пакета помилок, що виправляється, для послідовного коду без яких-небудь обмежень дорівнює $b' = j*b - (b - 1)$ для коду Ріда-Соломона поля Галуа $GF(2^b)$ з параметром чергування j.

Схему коду Ріда-Соломона з символьним чергуванням можна одержати зі схеми вихідного коду, вставляючи додатково до кожного елемента пам'яті j-1 елементів. Наприклад, для поля $GF(2^3)$ при чергуванні з параметром j=5 кожну тріаду елементів пам'яті потрібно замінити п'ятьма послідовно включеними тріадами.

Коди Ріда – Соломона, що виправляють подвійні помилки

Розглянемо побудову коду Ріда-Соломона, що виправляє подвійні помилки (тут, як і в попередньому випадку, під помилкою маємо на увазі не один помилковий двійковий символ, а елемент поля Галуа, тобто кілька двійкових символів).

Породжувальний поліном для s=2 містить чотири співмножники

$$RS(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4) =$$
$$=x^4 + x^3(z^4+z^2+z^3+z) + x^2(z^7+z^6+z^4+z^3)+ x(z^9+z^8+z^7+z^6) + z^{10}$$

$$(x-\alpha)(x-\alpha^2) = x^2 + x(z^2+z) + z^3$$

$$(x-\alpha^3)(x-\alpha^4) = x^2 + x(z^4+z^3) + z^7$$

$$
\begin{aligned}
RS(x) = \quad & x^2 + x(z^2+z) + z^3 \\
* \ & \underline{x^2 + x(z^4+z^3) + z^7} \\
& z^7x^2 + x\,(z^9+z^8) + z^{10} \\
& + x^3(z^4+z^3) + x^2(z^6+z^4) + x^4 + x(z^7+z^6) \\
& \underline{+ x^4 + x^3(z^2+z) + x^2z^3} \\
& \quad x^2z^7 + x\,z^9+x\,z^8+z^{10} \\
& + x^3z^4+x^3z^3+x^2z^6+x^2z^4+x\,z^7+ x\,z^6 \\
& \underline{+ x^4 + x^3z^2+x^3z + x^2z^3} \\
& \quad x^4 + x^3(z^4+z^2+z^3+z) + x^2(z^7+z^6+z^4+z^3) + x(z^9+z^8+z^7+z^6) + z^{10}
\end{aligned}
$$

Даний породжувальний поліном є справедливим для кожного поля Галуа.

Для обчислення породжувального полінома для конкретного поля необхідно обчислити відповідні коефіцієнти при псевдозмінних . Для цього кожен відповідний коефіцієнт у загальній формі необхідно розділити на примітивний поліном p(z). Залишок від ділення і буде являти собою шуканий коефіцієнт.

Приклад: Нехай код Ріда-Соломона, що виправляє подвійну помилку, будується для поля GF(16), побудованого як розширення поля GF(2) над примітивним поліномом $p(z) = z^4 + z + 1$ В цьому випадку довжина коду n=15 двійкових тетрад, кількість перевірочних символів p=4 тетради, кількість інформаційних символів k=11 тетрад. Побудова (15, 11)-коду Ріда-Соломона починаємо зі спрощення породжувального полінома, отриманого для загального випадку. Спростити цей поліном можна, розділивши кожний з його коефіцієнтів на породжувальний поліном поля Галуа.

1. $z^4+z^2+z^3+z$ | $z^4 + z + 1$

   $\underline{- z^4 + z + 1}$ | 1

   $z^3 + z^2 + 1$

2. $z^7+z^6+z^4+z^3$ | $z^4 + z + 1$

   $\underline{- z^7+z^4+z^3}$ | z^3+z^2

   z^6

   $\underline{- z^6+z^3+z^2}$

   z^3+z^2

3. $z^9+z^8+z^7+z^6$ | $\underline{z^4 + z + 1}$

   $\underline{- z^9+z^6+z^5}$ | $z^5+z^4+z^3$

   $z^8+z^7+z^5$

   $\underline{- z^8+z^5+z^4}$

   z^7+z^4

   $\underline{- z^7+z^4+z^3}$

   z^3

4. z^{10} | $z^4 + z + 1$

$- z^{10}+z^7+z^6$ | $z^6 + z^3 + z^2 + 1$

z^7+z^6

$- z^7+z^4+z^3$

$z^6+z^4+z^3$

$- z^6+z^3+z^2$

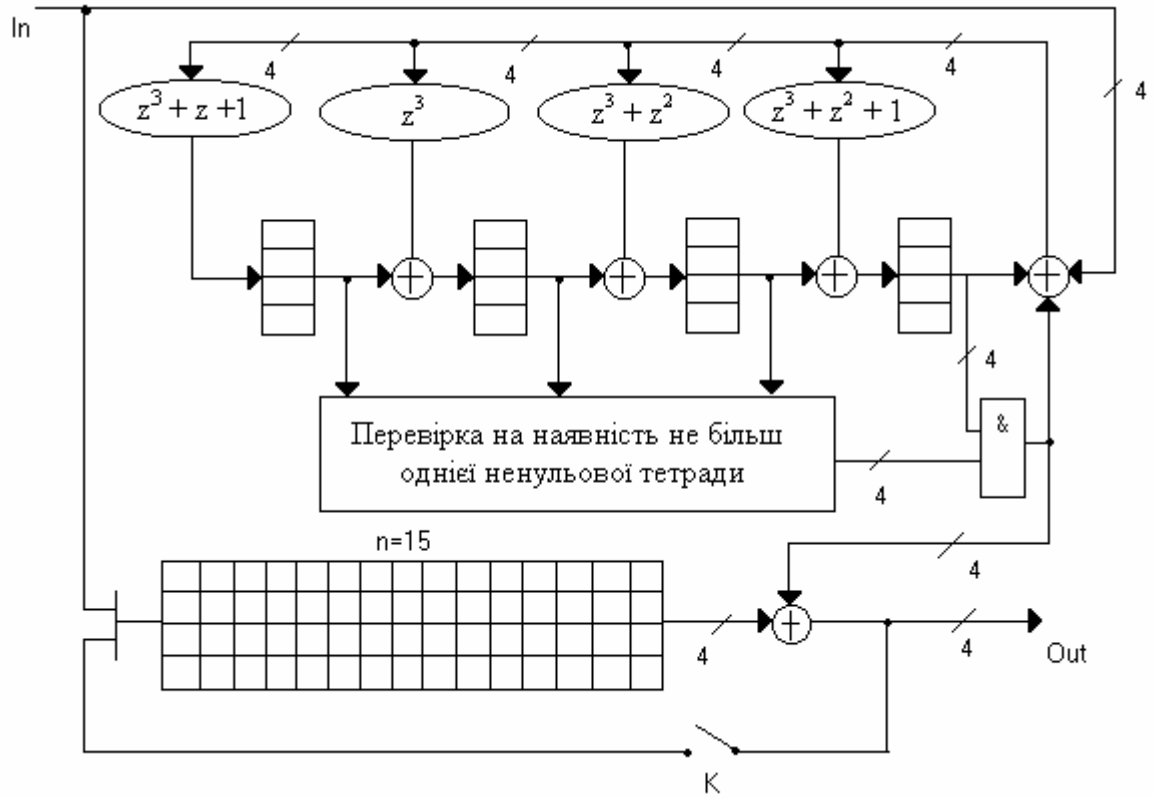
z^4+z^2

$- z^4+z + 1$

$z^2 + z + 1$

Таким чином породжувальний поліном коду Ріда-Соломона, що виправляє подвійні помилки, для поля GF(16):

$$RS(x) = x^4 + x^3(z^3+z^2+1) + x^2(z^3+z^2) + x\,z^3 + z^2 + z + 1$$



Декодер коду Ріда – Соломона аналогічний декодеру коду БЧХ, що виправляє подвійні помилки.

Декодер працює 3n тактів:

У 1-і n тактів формується синдром, кодове слово заноситься в буферний регістр

В 2-і n тактів виконується виправлення однієї з двох можливих помилкових тетрад. Синдром модифікується, кодове слово заново записується в буферний регістр.

У 3-і n тактів виконується виправлення другої помилкової тетради.

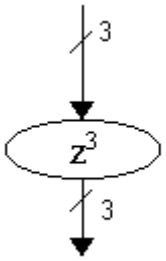
Даний код дозволяє виправити:

- для паралельного коду будь-які дві помилкові тетради;
- для послідовного коду будь-який пакет довжини 5 (без обмеження на характер розташування помилок), пакет помилок довжини 8 або 2 пакети довжини 4 (з обмеженням на характер розташування помилок).

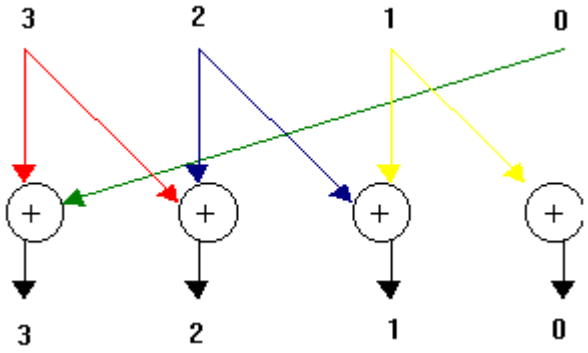
Аналогічно кодам Ріда – Соломона, що виправляє одиночну помилку, перетворюються помножувачі на константу.

Наприклад, для побудови схеми множення на константу z^3 необхідно знайти залишок від ділення $(a_3z^3 + a_2z^2 + a_1z + a_0)*z^3$ на примітивний поліном p(z)

$$
\begin{array}{l|l}
a_3z^6 + a_2z^5 + a_1z^4 + a_0z^3 & \underline{z^4 + z + 1} \\
\underline{- a_3z^6 + a_3z^3 + a_3z^2} & a_3z^2 + a_2z + a_1 \\
a_2z^5 + a_1z_4 + (a_3+a_0)z^3+a_3z^2 & \\
\underline{- a_2z^5 + a_2z^2 + a_2z} & \\
a_1z^4 + (a_3 +a_0)z^3 + (a_3 +a_2)z^2 + a_2z & \\
\underline{- a_1z^4 + a_1z + a_1} & \\
(a_3 +a_0)z^3 + (a_3 +a_2)z^2 + (a_2+ a_1)z + a_1 &
\end{array}
$$



По отриманому залишку від ділення схема помножувача на константу перетвориться для двійкової логіки:



Для збільшення коригувальних здібностей можна використовувати символьне чергування. Наприклад, при параметрі чергування j=2 одержимо код Ріда – Соломона (30, 22), що може виправляти множину різних помилок, зокрема, будь-який пакет довжини 13 у послідовному коді двійкових символів.

ЗГОРТАЛЬНІ КОДИ

Згортальні коди – один із самих популярних класів кодів, що виправляють помилки. Ці коди знайшли широке застосування на практиці: радіозв'язок (IMT-2000, GSM, IS-95), цифровий наземний і супутниковий зв'язок, радіомовні системи і т.д.

Коди Івадарі

Згортальні коди належать до класу деревоподібних. На відміну від блокових кодів, кодове слово деревоподібного коду n_0 (кадр кодового слова) формується в залежності не тільки від самого інформаційного блоку k_0 (інформаційного кадру), але також від m уже переданих інформаційних кадрів. Величина $k=(m+1)k_0$ називається інформаційною довжиною слова. Кодова довжина блоку $n=(m+1)n_0$ – це довжина кодового слова, на якій зберігається вплив одного кадру інформаційних символів.

Нехай λ і n_0 – будь-які позитивні числа. По визначенню кодом Івадарі називається виправляючий пакети помилок двійковий систематичний згортальний код з породжувальною $((n_0 - 1) \times n_0)$-матрицею з поліномів:

$$\mathbf{G}(x) = \begin{bmatrix} 1 & & & g_1(x) \\ & 1 & & g_2(x) \\ & & \ddots & \vdots \\ & & 1 & g_{(n_0-1)}(x) \end{bmatrix}$$

де для матричних елементів $g_{in_0}(x)$ використовувалося скорочене позначення $g_i(x)$ і $g_i(x) = x^{(\lambda+1)(2n_0-i)+i-3} + x^{(\lambda+1)(n_0-i)-1}$, $i = 1, \ldots, n_0 - 1$...

Найбільший степінь, рівний $(\lambda+1)(2n_0-1)-2$ має поліном $g_1(x)$. Таким чином, коди Івадарі є згортальними $((m+1)n_0, (m+1)(n_0-1))$ кодами, що виправляють пакети помилок довжини не більш λn_0, з числом кадрів $m=(\lambda+1)(2n_0-1)-2$.

Оскільки $n_0-k_0=1$ для кодів Івадарі одержуємо тільки один синдромний поліном

$$s(x) = \sum_{i=0}^{n_0-1} g_i(x)e_i(x) + e_{n_0}(x) =$$

$$= e_{n_0}(x) + \sum_{i=0}^{n_0-1} \left[x^{(\lambda+1)(n_0-i)-1} + x^{(\lambda+1)(2n_0-i)+i-3}\right] e_i(x).$$

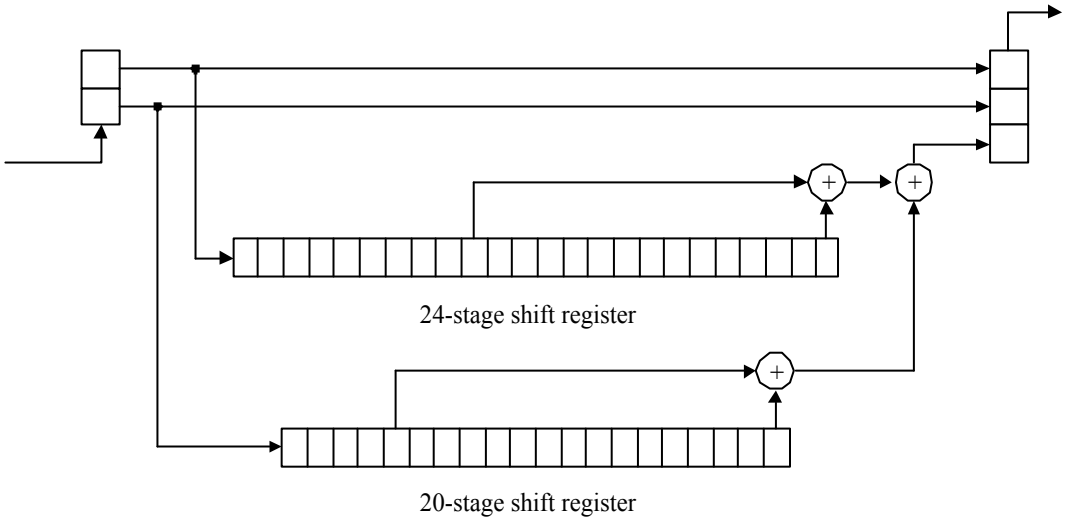
Побудова кодуючих пристроїв для кодів Івадарі, як правило, не викликає ускладнень. Декодування найкраще пояснити на прикладі, оскільки структура декодерів цих кодів однакова.

Приклад. Нехай $n_0=3,\ \lambda=4$.

У цьому випадку $m+1=24$, одержуємо (78, 24)-код Івадарі, що виправляє пакети помилок довжини 12.

Кодер коду реалізується на основі регістрів зсуву, кожний з яких виконує операцію множення на відповідний породжувальний поліном:

$g_1(x)=x^{23}+x^9$ і $g_2(x)=x^{19}+x^4$.



Для побудови схеми декодера необхідно спочатку виконати аналіз можливих синдромів для (78, 24)-коду Івадарі.

Якщо позначити через $c_3(x)$ поліном перевірочних символів, через $c_1(x)$ і $c_2(x)$ поліноми інформаційних символів, тоді відповідними поліномами помилок будуть поліноми $e_1(x)$, $e_2(x)$ і $e_3(x)$, а синдромний поліном дорівнює

$$s(x)=e_3(x)+(x^4+x^{19})e_2(x)+(x^9+x^{23})e_1(x).$$

Для пакета помилок, що починається в першому кадрі,

$$e_3(x)=e_{30}+e_{31}x+e_{32}x^2+e_{33}x^3,$$

$$e_2(x)=e_{20}+e_{21}x+e_{22}x^2+e_{23}x^3+e_{24}x^4,$$

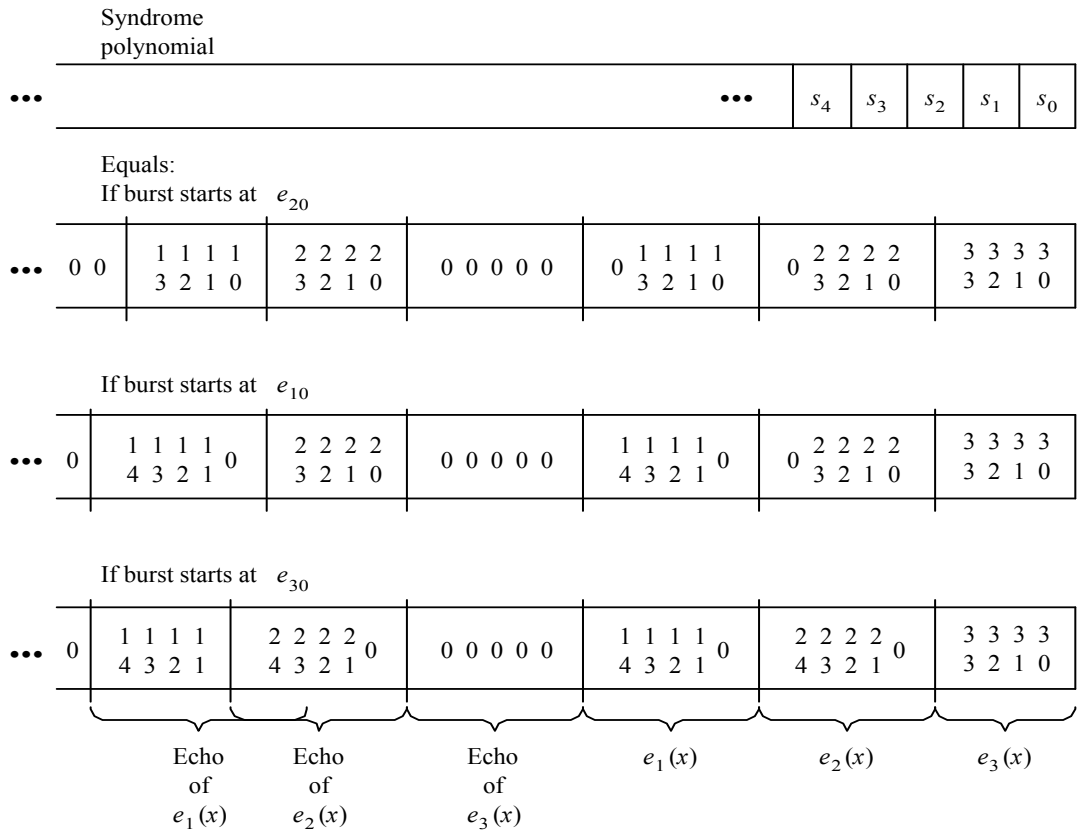
$$e_1(x)=e_{10}+e_{11}x+e_{12}x^2+e_{13}x^3+e_{14}x^4.$$

Виходячи з того, що довжина пакета помилок, що виправляється, не більше 12, можна затверджувати наступне. Якщо пакет починається в першому кадрі, то $e_{34}=0$. Якщо e_{24} не дорівнює нулеві, то e_{20} і e_{10} дорівнюють нулеві. Якщо e_{14} відмінний від нуля, то $e_{10}=0$.

На рисунку представлені коефіцієнти $s(x)$ для трьох випадків, у яких пакет помилок починається в $e_{10}(x)$, $e_{20}(x)$ і $e_{30}(x)$ відповідно. При формуванні синдрому поліноми перемежені і розташовуються в порядку

$e_1(x)$, $e_2(x)$ і $e_3(x)$, над їх луною потім виконується зворотне перемеження і вони з'являються в зворотному порядку $e_3(x)$, $e_2(x)$ і $e_3(x)$.

Кожен біт полінома $e_2(x)$ впливає на синдром двічі: за його першою появою випливає відгук із затримкою на 15 біт. Аналогічно кожен біт полінома $e_1(x)$ впливає на синдром двічі: за його першою появою випливає відгук із затримкою на 14 біт. Тому що e_{24} і e_{10} не можуть одночасно бути ненульовими, ці пари помилок ніколи не перекриваються. Кожен біт полінома $e_3(x)$ впливає на синдром один раз; за його першою появою випливають тільки нулі - один із затримкою на 14 біт, а інший з затримкою на 15 біт. Ці нулі визначають $e_3(x)$.



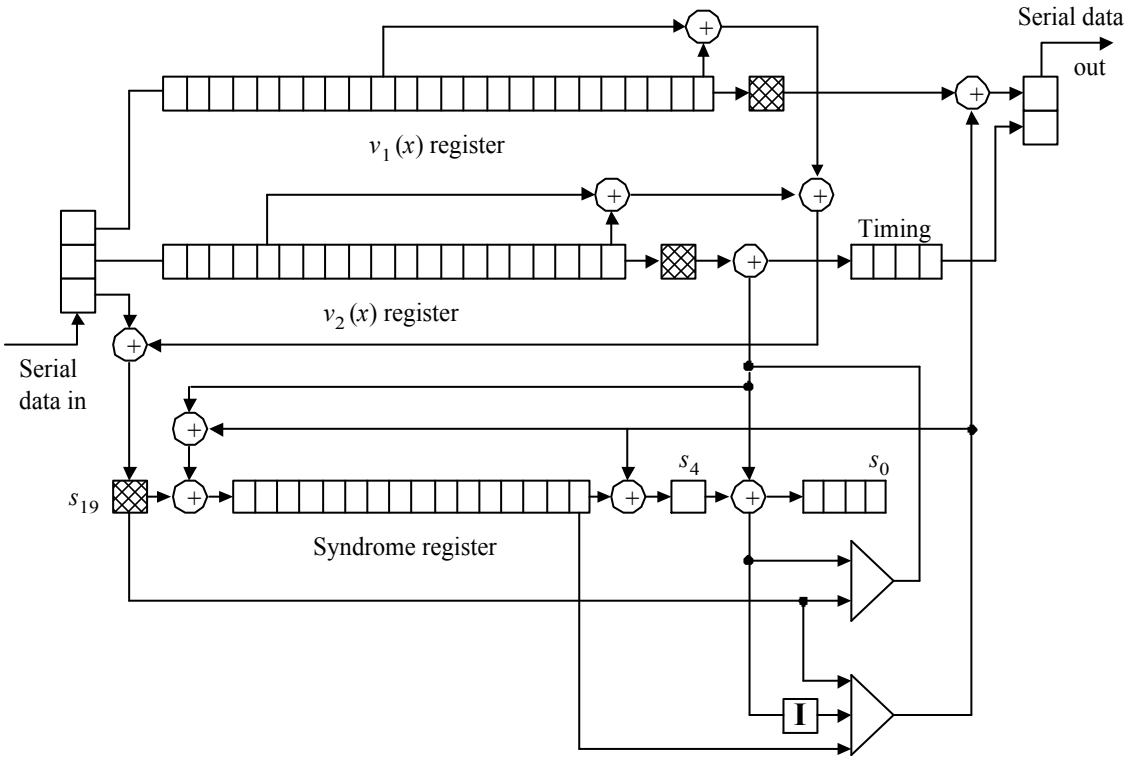
Якщо в дійсності пакет помилок починається в *l-му* кадрі, то синдром зсувається вправо на l біт і перед ним вставляється l нулів. Процедура декодування побудована таким чином, що ця конфігурація не буде сприйматися як пакет помилок, що починається в першому кадрі.

Перейдемо до опису декодера, зображеного на рисунку. Після того, як у нього надійшли перші 20 кадрів, у регістрі синдрому зберігаються перші 20 бітів синдрому, причому s_0 розташовується праворуч. Аналогічно перший прийнятий біт полінома $v_2(x)$ знаходиться в крайньому правому розряді v_2(x)-регістру. Протягом ще чотирьох наступних тактів перший прийнятий біт $v_1(x)$ не досягне крайнього правого розряду v_1(x)-регістра. Опис роботи декодера почнемо з цього моменту часу. Перші чотири біти синдрому залежать лише від помилок у

прийнятих перевірочних бітах і нас не цікавлять. Отже, та частина рисунка, що нанесена штриховими лініями, може бути опущена. Заштриховані розряди регістра зсуву також можуть бути опущені, а s_{19} може вводитися безпосередньо з попереднього суматора за модулем 2.

Спочатку декодер виправляє другий біт кожного кадру пакета помилок, а потім повертається до початку пакета і виправляє перший біт кожного кадру. Тому декодер має чотирирозрядний регістр зсуву, у якому він зберігає $з_2(x)$ після завершення виправлення. Третій біт кожного кадру є перевірочним, він не виправляється.

Для виправлення помилки в $v_2(x)$ декодер перевіряє компоненти s_4 і s_{19} синдрому. Якщо обидві компоненти дорівнюють одиниці, то правий біт $v_2(x)$ є помилковим і в потоці даних, а також у регістрі синдрому робляться відповідні виправлення. Пакет помилок $e_3(x)$, *що* починається в більш пізньому кадрі, не може викликати збою, тому що відгук $e_3(x)$ дорівнює нулю. Паралельно з цим для перебування помилки в $v_1(x)$ декодер чотирма кадрами пізніше перевіряє компоненти s_5 і s_{19} синдрому. Якщо вони обидві дорівнюють одиниці, а s_4 дорівнює нулю, то правий біт $v_1(x)$-регістру помилковий. Така перевірка $v_1(x)$ не приводить до відшукання помилок протягом перших чотирьох тактів, але потім починає виправляти помилки в $v_1(x)$. Пакет $e_3(x)$, *що* починається в більш пізньому кадрі, не може викликати збою, тому що відгук $e_3(x)$ дорівнює нулю. У силу вибору способу перевірки ще не виправлений біт $e_2(x)$ також не може викликати збою, і тому s_4 дорівнює нулю.



Після появи пакета помилок довжини 12 прийняте слово не повинне містити більше помилок, щоб синдромові ніщо не заважало під час виправлення цього пакета. Для цього необхідно, щоб до 29-го кадру не відбувалося помилок. Отже, між пакетами помилок повинно бути вільно від помилок не менш 24 кадрів (72 правильних біта). При виконанні цього правила декодер буде успішно виправляти пакети помилок довжини не більше 12, що надходять друг за другом.

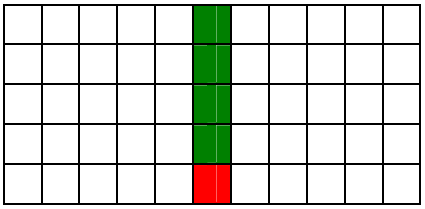
Векторні коди

Векторні коди (orchard code – код фруктового саду) – це клас кодів з контролем парності. Ці коди дозволяють підвищити надійність виявлення і виправлення помилок при мінімальному підвищенні надмірності. Векторний метод передбачає використання декількох векторів двійкових символів для обробки даних, представлених у виді паралельного потоку. Він набагато більш ефективний, чим звичайні способи контролю парності.

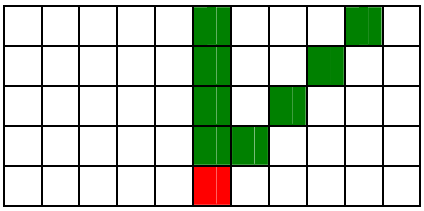
Концепція векторного коду заснована на тому, що місце перебування будь-якого помилкового біта можна однозначно визначити на перетинанні двох не тільки перпендикулярних векторів, але також обраних спеціальним способом. При цьому для опису контрольних векторів коду використовується графічне представлення, що пояснює принцип роботи і на підставі якого будуються кодуючі та декодуючі пристрої.

Розглянемо кілька варіацій векторного коду, з яких можна виділити наступні.

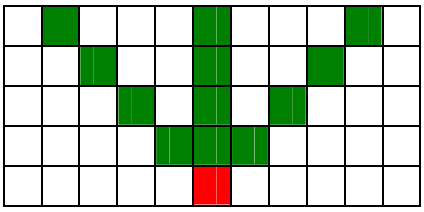
Код за паритетом або код з парним числом одиниць, є блоковим кодом, містить один перевірочний символ і дозволяє виявляти всі помилки непарної кратності. Формування перевірочного символу в паралельному потоці двійкових символів можна представити в такий спосіб (червоним позначений перевірочний символ, що формується сумою за модулем два зелених інформаційних).



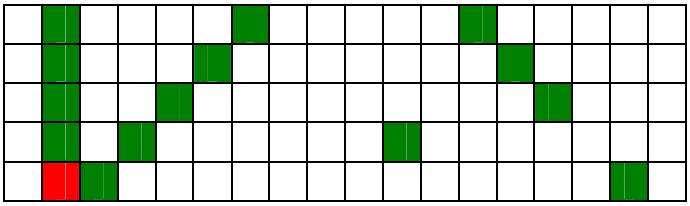
Якщо перевірочний символ формувати за допомогою суми за модулем два двох векторів, то з'являється можливість не тільки виявляти, але і виправляти одиночні помилки:



Можливий також і інший варіант формування контрольного символу (який для виправлення одиночних помилок має зайву надмірність у порівнянні з попереднім):



І, нарешті, основний варіант формування перевірочного символу має такий вигляд:



У типовій тривекторній схемі контролю за методом фруктового саду пристрій кодування складається з паралельної регістрової матриці і генератора сигналів парності. Вихід генератора сигналів парності по зворотному зв'язку надходить на вхід регістрової матриці для формування перевірочного біта. Декодер виконаний аналогічно кодеру з тим виключенням, що, оскільки надійшовший на його вхід паралельний потік даних уже містить перевірочний біт, вихід генератора парності надходить на вхід синдромного послідовного регістра. Перший коригувальний каскад складається з паралельної регістрової матриці, синдромного регістра і блоку розпізнавання коду. Схема розпізнавання коду є основним елементом, яким розрізняються наступні коригувальні каскади. Для тривекторного коду може бути два таких каскади.

Контроль і виправлення помилок за векторним методом можна робити з використанням більш ніж трьох векторів. Для випадку n векторів необхідно мати (n-1) коригувальних каскадів. Перший коригувальний каскад буде виправляти всі помилки, для яких у синдромний регістр буде надходити n ознак помилок. Кожен наступний каскад буде виправляти помилки, для яких у синдромний регістр будуть надходити ознаки помилок , число яких на одиницю менше, ніж для попереднього каскаду.

Останній коригувальний каскад буде виправляти всі помилки, для яких буде по дві ознаки помилок.

До безсумнівного достоїнства векторних кодів є їхня наочність, що забезпечує їхнє графічне представлення. Разом з тим, хоча по своїй природі вони є згортальними, немає ніякого згадування про існування таких кодів. Крім того, немає ніякого алгоритму вибору векторів і, у тому числі, відстані між ними. Для характеристики коригувальних можливостей кодів використовуються такі вираження, як "звичайно" виявляють або виправляють помилки або "багато" помилок, що виявляються і виправляються.

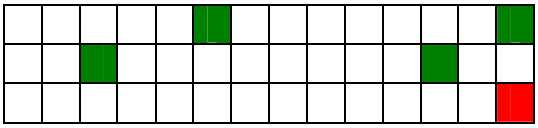
Графічне представлення кодів Івадарі

Розглянемо графічне представлення декількох варіантів кодів Івадарі.

1) Нехай конструктивний параметр $n_0 = 3$.

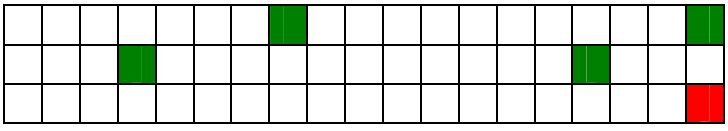
Для $\lambda = 2$ одержуємо два породжувальних полінома: $g_1(x) = x^{13} + x^5$, $g_2(x) = x^{11} + x^2$ і виправлюваний пакет помилок $\lambda n_0 = 6$.

У графічному виді:



Для $\lambda = 3$ одержуємо два породжувальних полінома: $g_1(x) = x^{18} + x^7$, $g_2(x) = x^{15} + x^3$ і виправлюваний пакет помилок $\lambda n_0 = 9$.

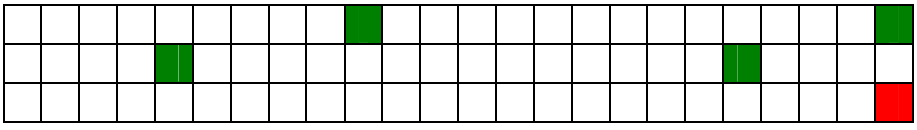
У графічному виді:



Для $\lambda = 4$ одержуємо два породжувальних полінома: $g_1(x) = x^{23} + x^9$, $g_2(x) = x^{19} + x^4$

і виправлюваний пакет помилок $\lambda n_0 = 12$.

У графічному виді:



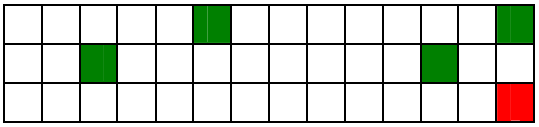
Отже, при $n_0 = 3$ перевірочний біт формується по 4-м (у загальному випадку – по $2(n_0 - 1)$) контрольним бітам, які можна розбити на два вектори. Відстань між кадрами, у яких знаходяться контрольні біти,

залежить від параметра λ. У першому векторі воно дорівнює λ, у другому - λ-1.

2) Нехай конструктивний параметр $\lambda = 2$.

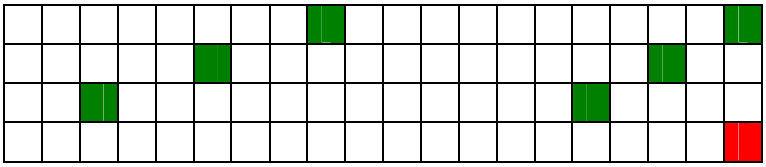
Для $n_0 = 3$ одержуємо два породжувальних полінома: $g_1(x) = x^{13} + x^5$, $g_2(x) = x^{11} + x^2$ і виправлюваний пакет помилок $\lambda n_0 = 6$.

У графічному виді:



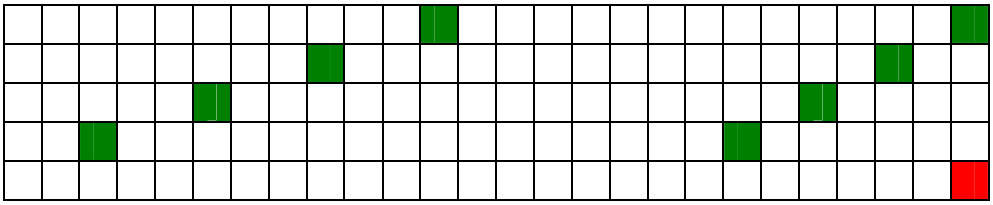
Для $n_0 = 4$ одержуємо три породжувальних полінома: $g_1(x) = x^{19} + x^8$, $g_2(x) = x^{17} + x^5$, $g_3(x) = x^{15} + x^2$ і виправлюваний пакет помилок $\lambda n_0 = 8$.

У графічному виді:



Для $n_0 = 5$ одержуємо чотири породжувальних полінома:
$g_1(x) = x^{25} + x^{11}$, $g_2(x) = x^{23} + x^8$,
$g_3(x) = x^{21} + x^5$, $g_4(x) = x^{19} + x^2$
і виправлюваний пакет помилок $\lambda n_0 = 8$.

У графічному виді:



Отже, розглянуті приклади показують залежність відстані між кадром з останнім контрольним бітом першого вектора і кадром з першим контрольним бітом другого вектора дорівнює $(n_0 + 2(\lambda - 1))$.

Графічний спосіб опису кодів Івадарі дає наочне представлення про принципи побудови цих кодів, дозволяє будувати ці коди, не використовуючи породжувальних поліномів, поняття яких вимагає знань з алгебри полів Галуа, а також виконати порівняльний аналіз з векторними кодами. У порівнянні з векторними кодами коди Івадарі мають наступні переваги: формалізований алгоритм побудови контрольних векторів для формування перевірочного символу, гарантоване виправлення всіх пакетів помилок довжини λn_0, декодування не вимагає декількох каскадів виправлення помилок.

ТЕОРІЯ ІНФОРМАЦІЇ

Інформаційні моделі сигналів

Всякий інформаційний пристрій працює за єдиною схемою. На вхід пристрою подається сукупність повідомлень x1,x2,...,xn (ансамблі подій). Завдання пристрою полягає в тому, щоб передати сукупність цих повідомлень на вихід Y з досить високою вірогідністю.

У процесі передачі повідомлення може піддаватися численним перетворенням, істотно змінювальним його фізичні характеристики. Однак, передана інформація повинна залишатися інваріантної при всіх перетвореннях. Кількість переданої одержувачеві інформації зв'язано з невизначеністю, що мала місце щодо переданого повідомлення. У зв'язку з цим необхідно ввести кількісну міру оцінки інформації і невизначеності переданих повідомлень.

Кількісна міра інформації

Всяка інформація виходить споживачем після прийняття повідомлення, тобто в результаті досліду. Якщо дослід має лише один результат і не містить ніякої невизначеності, то спостерігач заздалегідь буде знати результат цього досліду. У результаті здійснення такого досліду спостерігач не одержить ніякої інформації. Нехай дослід має два рівноімовірних результати. Результат контролю, наприклад, повинний указати, що контрольований параметр знаходиться в межах норми або за її межами. Передане повідомлення в цьому випадку може приймати два значення і містить визначену інформацію.

Розглянемо загальний випадок, коли джерело може передавати незалежні і несумісні повідомлення x1, x2,..., xn з ймовірностями p(x1),p(x2),...,p(xn) відповідно. Природно, що чим менше апріорна імовірність події, тим більша кількість інформації воно несе. Так, наприклад, повідомлення про те, що влітку температура повітря в Криму вище нуля, не несе істотної інформації, тому що імовірність такої події дуже велика. Повідомлення ж про те, що температура повітря в Криму в червні нижче нуля, містить значно більше інформації, тому що така подія є досить рідкою.

Таким чином, природно припустити, що ***кількісною мірою невизначеності є величина, зворотна його апріорної імовірності.***

$$I(x_i) = \log_a 1/P(x_i)$$

Дана міра має *властивість адитивності*:

$$\frac{1}{P(x_i)*P(x_j)} \neq \frac{1}{P(x_i)} + \frac{1}{P(x_j)}$$

$$I(x_i, x_j) = \log_a \frac{1}{P(x_i)P(x_j)} = \log_a \frac{1}{P(x_i)} + \log_a \frac{1}{P(x_j)} = I(x_i) + I(x_j)$$

Як видно, ця міра у випадку подій з однаковим результатом дає нульове значення кількості інформації.

Вираження для $I(x_i)$ характеризує *кількість інформації, що* утримується в повідомленні x_i. Воно характеризує також апріорну невизначеність цього повідомлення і може бути використане для кількісної оцінки невизначеності повідомлення.

$$H(x_i) = \log_a \frac{1}{P(x_i)}$$

Цю величину, що характеризує невизначеність окремого (і-го) повідомлення, прийнято називати ***частинною ентропією***.

$$I(x) = -\sum_{i=1}^{n} p(x_i) * \log_2 p(x_i)$$

$$H(x) = -\sum_{i=1}^{n} p(x_i) * \log_a p(x_i)$$

Незважаючи на збіг залежностей, ентропія H(x) і кількість інформації I(x) принципово різні. Ентропія H(x), що виражає *середню невизначеність стану джерела повідомлень* є об'єктивною характеристикою джерела повідомлень, і, якщо відомо статистику повідомлень, може бути обчислена апріорно, тобто до одержання повідомлення, I(x) є апостеріорною характеристикою і *визначає кількість інформації, одержувану з надходженням повідомлень.* H(x) є міра недоліку інформації про стан окремої системи. З надходженням інформації про стан системи ентропія останньої знижується. Збіг виразу для I(x) з виразом для H(x) свідчить лише про те, що кількість одержуваної інформації дорівнює чисельно ентропії, що мала місце щодо джерела повідомлень.

Одиниці виміру кількості інформації й ентропії залежать від вибору основи логарифма у формулах.

a=10 – ***діти*** (десяткові одиниці)
a=2 – ***біти*** (двійкові одиниці)
a=e – ***ніти*** (натуральні одиниці)

Міра кількості інформації в наведеному виді вперше була запропонована Клодом Шенноном у 1948 році.

Властивості безумовної ентропії дискретних повідомлень

$$H(x) = -\sum_{i=1}^{n} p(x_i) * \log_a p(x_i)$$

Ця формула виражає ентропію дискретних повідомлень (ентропію Шеннона, безумовну ентропію).

Ентропія Шеннона має наступні *властивості*:

1. Ентропія є величина дійсна, обмежена і ненегативна. Це випливає з формули, тому що імовірності $p(x_i)$ є величини ненегативні, ув'язнені в проміжку [0,1]
2. Ентропія детермінірованих повідомлень дорівнює нулеві.
3. Ентропія максимальна, якщо повідомлення рівноімовірні.

У випадку n рівноімовірних повідомлень їхньої імовірності

$$p(x_1)=p(x_2)=...=p(x_n)=1/n$$

При цьому ентропія повідомлень буде дорівнювати

$$\mathbf{H(x)_{max}=\log_2 n}$$

Як видно з цього виразу, у випадку рівноімовірних подій ентропія зростає зі збільшенням кількості подій n.

4. Ентропія системи двох альтернативних подій може змінюватися в межах від нуля до одиниці.

$p(x_2) = 1 – p(x_1)$

$H(x) = -p(x_1)*\log_2 p(x_1) – p(x_2)*\log_2 p(x_2) = p(x_1)*\log_2 p(x_1) – (1 - p(x_1))*\log_2 p(x_2)$

З останнього вираження видно, що ентропія дорівнює нулеві при

$p(x_1) = 0; p(x_2) = 1,$

або $p(x_1) = 1; p(x_2) = 0.$

Максимум ентропії буде мати місце, коли

$p(x_1) = p(x_2).$

При цьому максимальне значення ентропії

$$H(x)_{\max} = -\frac{1}{2}\log_2 \frac{1}{2} - \frac{1}{2}\log_2 \frac{1}{2} = 1 \text{ біт}$$

Таким чином, можна затверджувати, що один біт (одна двійкова одиниця) - це ентропія системи двох рівноімовірних незалежних подій.

Ентропія складних повідомлень. Ентропія об'єднання, умовна ентропія

При рішенні задач передачі інформації часто мають справу з декількома джерелами, що дають залежні повідомлення. *Сукупність повідомлень, вироблюваних декількома джерелами, називають* ***складним повідомленням.***

Нехай є два джерела повідомлень. Повідомлення першого джерела приймають значення $x_1,x_2,...,x_n$ з ймовірностями $p(x_1),p(x_2),...,p(x_n)$, а повідомлення другого джерела приймають значення $y_1,y_2,...,y_n$ з ймовірностями

(y_1),p(y_2),...,p(y_m). Спільну ентропію сукупності повідомлень X і Y (***ентропію об'єднання***) можна представити у виді:

$$H(X,Y) = -\sum_{i=1}^{n}\sum_{j=1}^{m} p(x_i, y_j)\log_2 p(x_i, y_j)$$

де p(x_i, y_j) - імовірність спільної появи повідомлень x_i і y_j.

Оскільки спільна імовірність може бути представлена у виді

$$p(x_i, y_i) = p(x_i) * p(\frac{y_j}{x_i})$$

де $p(y_j/x_i)$ - умовна імовірність повідомлення y_j за умови, що надійшло повідомлення x_i, у результаті математичних перетворень вираження для ентропії можна представити у виді:

H(X,Y) = H(X) + H(Y/X)

або

H(X,Y) = H(Y) + H(X/Y)

Основний зміст умовної ентропії H(Y/X) полягає в тому, що вона показує, яку ентропію дають повідомлення Y, коли уже відома ентропія повідомлень X.

Таким чином, *спільна ентропія двох повідомлень дорівнює сумі безумовної ентропії одного з повідомлень і умовної ентропії другого повідомлення.*

Основні властивості ентропії складних повідомлень (ентропії об'єднання)

1. При *статистично незалежних повідомленнях* X і Y *спільна ентропія* дорівнює сумі ентропій повідомлень
   H(X,Y)= H(X) + H(Y),
   тому що в цьому випадку H(Y/X) = H(Y).
2. При *повній статистичній залежності повідомлень* X і Y *спільна ентропія* дорівнює безумовної ентропії одного з повідомлень, тому що в цьому випадку
   H(Y/X)=0 і H(X/Y)=0;
   H(X,Y)= H(X) = H(Y).
3. *Умовна ентропія* може змінюватися в межах
   $0 \le H(Y/X) \le H(Y)$.
4. Для *спільної ентропії* завжди справедливе співвідношення
   $H(X,Y) \le H(X) + H(Y)$.

Навчальне видання

КОНСПЕКТ ЛЕКЦІЙ
З КУРСУ
“ТЕОРІЯ КОРИГУЮЧИХ КОДІВ”

(для студентів спеціальності 7.091502 “Системне програмування”)

Укладач: Дяченко Олег Миколайович

Литература

1. Richard E.Blahut.Algebraic Codes for Data Transmission/ Cambridge University Press, 2012. – 498 p.
2. Richard E. Blahut. Theory and Practice of Error Control codes /Addison-Wesley Publishing Company, 1986.– 576 p.
3. Peterson W.W., Weldon E.J.,Jr. Error-correcting codes.- 2nd ed.- Cambridge (Mass.): MIT Press., 1971.– 595 p.
4. Robert H. Morelos-Zaragoza. The art of error correcting coding/ SONY Computer Science Laboratories, Inc. JAP, John Wiley & Sons, Ltd, 2002. – 219p.
5. Тарасенко А.Н., Дяченко О.Н. Достоверность обнаружения кратных ошибок устройствами сигнатурного анализа/ Ред. журн. "Электрон. моделирование".- Киев, 1987. - 13 с.: ил. - Деп. в ВИНИТИ 06.04.87. N 2446-В87.
6. Дяченко О.Н. Аналитический метод вычисления сигнатур /Донецк. политехн. ин-т.- Донецк,1989. - 16 с.- Деп. в УкрНИИНТИ 1989, N 585-Ук89.
7. Дяченко О.Н., Львов Г.М., Тарасенко А.Н. Достоверность обнаружения неисправностей сигнатурным анализатором и контрольным суммированием //Проблемы диагностирования микропроцессорных систем: Тез. докл. науч.-техн. конф. - Ужгород, 14-18 окт. 1987. Киев, 1987. - С.32.
8. Дяченко О.Н., Тарасенко А.Н. Быстродействующий алгоритм аналитического вычисления сигнатур //Автоматизация контроля вычислительных устройств и систем: Тез. докл. науч.-техн. конф. Винница, 19-21 окт. 1988. - Киев, 1988. - С.141.
9. Дяченко О.Н., Тарасенко А.Н. Проектирование комбинационных схем с учетом требований сигнатурной тестируемости //Вопросы разработки вычислительной техники: Тез. докл. науч.-техн. конф. Кишинев, 16-17 мая 1989.- Кишинев, 1989. - С.9.
10. А.С. 1311011 СССР, МКИ4 H03K 13/00, G06F 11/00. Логический анализатор / О.Н.Дяченко (СССР) № 4016159/24-24; Опубл. 15.05.87.- Бюл. № 18.
11. А.С. 1336010 СССР, МКИ4 G06F 11/16. Многовходовый сигнатурный анализатор /А.Н.Тарасенко, Г.М.Львов, О.Н.Дяченко, А.И.Уткин, А.А.Коновалов, Н.Л.Антипова (СССР) № 4053868/24-24; Опубл. 07.09.87. - Бюл. № 33.
12. А.С. 1383363 СССР, МКИ4 G06F 11/00. Сигнатурный анализатор / А.Н.Тарасенко, Г.М.Львов, О.Н.Дяченко, А.И.Уткин, Н.Л.Антипова (СССР) № 4158907/24-24; Опубл. 23.03.88.- Бюл. № 11.
13. А.С. 1430956 СССР, МКИ4 G06F 11/16. Многоканальный сигнатурный анализатор /А.Н.Тарасенко, Г.М.Львов, О.Н.Дяченко, А.И.Уткин, Н.Л.Антипова, Г.В.Кунашев (СССР) № 4236241/24-24; Опубл. 15.10.88.- Бюл. № 38.

14. А.С. 1552184 СССР, МКИ5 G06F 11/00. Устройство для контроля цифровых узлов /А.Н.Тарасенко, О.Н.Дяченко, В.К.Майдыковский, Н.А.Зосимова (СССР) № 4462267/24-24; Опубл. 23.03.90.- Бюл. № 11.
15. А.С. 1649558 СССР, МКИ5 G06F 13/00, 11/00. Устройство для сопряжения абонента с общей магистралью /О.Н.Дяченко, Я.В.Юхновецкий, В.Н.Гавриш (СССР) № 4686956/24; Опубл. 15.05.91.- Бюл. № 18.
16. А.С. 1645958 СССР, МКИ5 G06F 11/00. Устройство для контроля цифровых узлов /А.Н.Тарасенко, О.Н.Дяченко, И.Н.Шаталов, А.И.Дойных, Н.А.Зосимова, А.В.Анцыгин (СССР) № 4677075/24, 4462267/24; Опубл. 30.04.91.- Бюл. № 16.
17. Дяченко О.Н.,Зосимова Н.А. Оценка эффективности способов сжатия информации при компактном тестировании цифровых устройств конвейерного типа /Вопросы специальной радиоэлектроники. Серия "Общие вопросы радиоэлектроники".- Москва,1989. - Выпуск 7. С.5-10.
18. Дяченко О.Н.,Тарасенко А.Н. Способы выделения диагностической информации из сигнатур //Электрон. моделирование. - 1990. 12, N 5. - С.64-70.
19. Дяченко О.Н. Анализ сигнатурной тестируемости комбинационных схем //Автоматика и вычислительная техника. - 1990. - N 5. С.85-89.
20. Дяченко О.Н., Тарасенко А.Н. Методы выделения диагностической информации из сигнатур //Проблемы автоматизации контроля электронных устройств: Тез. докл. науч.-техн. конф. - Винница, 1315 нояб. 1990. - Киев, 1990. - Вып. 1. - С.45-46.
21. Дяченко О.Н., Шаталов И.Н. Способ контроля и коррекции кодов при передаче и хранении информации //Проблемы автоматизации контроля электронных устройств: Тез. докл. науч.-техн. конф. Винница, 13-15 нояб. 1990. - Киев, 1990. - Вып. 1. - С.46-47.
22. Дяченко О.Н., Тарасенко А.Н. Сигнатурно-синдромное тестирование комбинационных схем // Проблемы автоматизации контроля электронных устройств: Тез. докл. науч.-техн. конф. - Винница, 13-15 нояб. 1990. - Киев, 1990. - Вып. 1. - С.56-57.
23. Дяченко О.Н. Тарасенко А.Н. Сигнатурно-синдромный анализатор // Проблемы автоматизации контроля и диагностирования сложных технических систем: Тез. докл. науч.-техн. конф. - Житомир, 17-19 сент. 1991 - К.: УкрНИИНТИ, 1991. - С.20-21.
24. Тарасенко А.Н., Дяченко О.Н., Гусев Б.С., Зинченко Ю.Е. Программно-аппаратные средства встроенного контроля и диагностики цифрового процессора обработки сигналов //Науч.-техн. конф. по завершенным НИР: Сборник тез. докл. науч.-техн. конф. (Донецк, февр. 1991). - Донецк: ДПИ, 1991. - Часть 2. - С.54.

25. Тарасенко А.Н., Гусев Б.С., Куприн В.М., Зинченко Ю.Е., Имас Т.А., Журавель А.П., Дяченко О.Н. Разработка тестера на базе микро-ЭВМ "Электроника-60" для контроля средств СОУИ //Науч.-техн. конф. по завершенным НИР: Сборник тез. докл. науч.-техн. конф. (Донецк, февр. 1991). - Донецк: ДПИ, 1991. - Часть 2. - С.52.
26. Тарасенко А.Н., Дяченко О.Н. Методические указания к выполнению лабораторных работ по курсу "Техническая диагностика и эксплуатация ЭВМ" (для студентов специальности 2201). - Донецк: ДПИ,1991. - 94 с.
27. Гусев Б.С., Ефремов С.С., Дяченко О.Н. Методические указания к выполнению лабораторных работ по курсу "Схемотехника ЭВМ". Раздел "Узлы ЦВМ" (для студентов специальности 2201).- Донецк: ДПИ, 1992.- 30с.
28. Дяченко О.Н., Махамат Алькер. Корреляционный метод компактного тестирования в области сигнатурного анализа/ Донецк. политехн. ин-т.- Донецк, 1992.- 15с.- Деп. в УкрИНТЭИ 07.05.92, N645-Ук92.
29. Дяченко О.Н., Лагутин А.А. Аппаратная реализация сигнатурно-синдромного тестирования комбинационных схем/ Донецк. политехн. ин-т.- Донецк, 1992.- 23с.- Деп. в УкрИНТЭИ 02.07.92, N992-Ук92.
30. Дяченко О.Н., Тарасенко А.Н. Спектральный метод компактного тестирования в области сигнатурного анализа// Электрон. моделирование.- 1992.- 14, N6.- С.60-65.
31. Тарасенко А.Н., Зинченко Ю.Е., Дяченко О.Н. Методические указания к выполнению лабораторных работ по курсу "Техническая диагностика и эксплуатация средств ВТ" (для студентов специальности 2201).- Донецк: ДПИ, 1992.- 53с.
32. Зинченко Ю.Е., Дяченко О.Н., Климова Т.В. Методические указания к выполнению лабораторных работ по курсу "Техническая диагностика и эксплуатация средств ВТ". Раздел "Сигнатурный анализ" (для студентов специальности 2201).- Донецк: ДПИ, 1993.- 43с.
33. Дяченко О.Н., Зинченко Ю.Е. Методические указания к выполнению лабораторных работ по курсу "Техническая диагностика и эксплуатация средств ВТ". Раздел "Методы компактного тестирования" (для студентов специальности 2201).- Донецк: ДПИ, 1993.- 32с.
34. А.С. 1737452 СССР, МКИ5 G06F 11/00. Сигнатурный анализатор/ А.Н.Тарасенко, О.Н.Дяченко (СССР) № 4744428/24; Опубл. 30.05.92.Бюл.№ 20.
35. Зинченко Ю.Е., Дяченко О.Н. Методические указания к выполнению лабораторных работ по курсу "Техническая диагностика и эксплуатация средств ВТ". Раздел "Моделирование компактного анализа на ПЭВМ" (для студентов специальности 2201).- Донецк: ДПИ, 1993.33с.

36. Зинченко Ю.Е., Дяченко О.Н. Методические указания к выполнению лабораторных работ по курсу "Техническая диагностика и эксплуатация средств ВТ". Раздел "Моделирование компактного тестирования на ПЭВМ" (для студентов специальности 2201) (на дискете).Донецк: ДПИ, 1993.- 40с.
37. Дяченко О.Н., Аль-Дахле Мухаммед Зайд. Сигнатурно-синдромное тестирование многовыходных комбинационных схем/ Донецк. политехн. ин-т.- Донецк, 1993.- 16с.- Деп. в ГНТБ Украины 20.07.93 N 1551-Ук93.
38. Дяченко О.Н., Третьяков А.С. Альтернативная аппаратная реализация спектрального метода компактного тестирования на основе коэффициентов арифметического спектра// Теоретическая и прикладная информатика: Тез. докл. науч.-техн. конф.- Донецк, 26-27 окт. 1993.- Донецк, 1993.- С.55-56.
39. А.С. 1797118 СССР, МКИ5 G06F 11/00. Многоканальный сигнатурный анализатор/ О.Н.Дяченко, А.П.Журавель (СССР) № 4752972/24; Опубл. 23.02.93.- Бюл. № 7.
40. А.С. 1829035 СССР, МКИ5 G06F 11/00. Сигнатурно-синдромный анализатор/ О.Н.Дяченко (СССР) № 4864016/24; Опубл. 23.07.93.Бюл. № 27.
41. А.С. 1837291 СССР, МКИ5 G06F 11/00. Многоканальный сигнатурный анализатор/ О.Н.Дяченко (СССР) № 4767976/24; Опубл. 30.08.93.- Бюл. № 32.
42. Дяченко О.Н. Методические указания к выполнению лабораторных работ по курсу "Теория информации и кодирования ". Раздел "Помехоустойчивое кодирование " (для студентов специальности 2201,6.0915).- Донецк: ДонГТУ, 1995.- 17с.
43. Дяченко О.Н., Волков В.Э. Компактный анализ с локализацией пакетов ошибок на основе кодов Файра// Доклады региональной научн. конф. "Творческое наследие В.И.Вернадского и современность". Секция 4 "Актуальные проблемы вычислительной техники, информатики и энергетики". Часть 1.- Донецк: ДонГТУ, 1995.- С.24-25.
44. Дяченко О.Н. Методы компактного тестирования цифровых устройств с локализацией пакетов ошибок // Доклады региональной научн. конф. "Творческое наследие В.И.Вернадского и современность". Секция 4 "Актуальные проблемы вычислительной техники, информатики и энергетики". Часть 1.- Донецк: ДонГТУ, 1995.- С.26-28.
45. Дяченко О.Н., Волков В.Э. Компактное тестирование цифровых схем с локализацией пакетов ошибок на основе укороченных кодов Файра/ Донец. гос. техн. ун-т.- Донецк,1995.- 13с.: ил.- Деп. в ГНТБ Украины 15.06.95 N 1519-Ук95.
46. Дяченко О.Н., Козицкий И.А. Эффективность компактного тестирования с локализацией ошибок// Автоматизация проектирования и производства изделий в машиностроении: Тез. докл. международной науч. -практ. конф.- Луганск,14-17 мая 1996.- Луганск, 1996.- С.124.

47. Дяченко О.Н., Козицкий И.А. Эффективность компактного тестирования комбинационных схем с локализацией ошибок/ Донец. гос. техн. ун-т.- Донецк, 1996.- 14 с.- Деп. в ГНТБ Украины 12.08.96 № 1614 – УК96.
48. Дяченко О.Н., Разицкий С.В. Компактное тестирование цифровых схем с локализацией пакетов ошибок/ Донец. гос. техн. ун-т.Донецк, 1996.- 19с.:ил.- Деп. в ГНТБ Украины 12.08.96 № 1613 – УК96.
49. Дяченко О.Н. Метод аналитического вычисления сигнатур// Сборник трудов факультета вычислительной техники и информатики. Выпуск 1. Донецкий государственный технический университет.- Донецк: ДонГТУ, 1996.- С.97-102.
50. Дяченко О.Н. Сравнительная оценка эффективности методов компактного тестирования комбинационных схем// Сборник трудов факультета вычислительной техники и информатики. Выпуск 1. Донецкий государственный технический университет.- Донецк: ДонГТУ, 1996.- С.103-110.
51. Патент України № 6840, МКИ5 G 06 F 11/00. Сигнатурно-синдромний аналізатор/ О.М.Дяченко (Україна); Опубл. 31.03.95.- Бюл. №1.
52. Патент України № 6922, МКИ5 G 06 F 11/00. Багатоканальний сигнатурний аналізатор/ О.М.Дяченко, О.П.Журавель (Україна); Опубл. 31.03.95.- Бюл. №1.
53. Патент України № 6877, МКИ5 G 06 F 11/00. Багатоканальний сигнатурний аналізатор/ О.М.Дяченко, (Україна); Опубл. 31.03.95.- Бюл. №1.
54. Дяченко О.Н. Компактное тестирование запоминающих устройств с локализацией пакетов ощибок// Электрон. моделирование.- 1996.- 18, №6.- С.43-48.
55. Дяченко О.Н., Герасимов А.Н. Компактное тестирование на основе четырехзначной логики// Информатика, кибернетика и вычислительная техника (ИКВТ - 97). Сборник научных трудов Донецкого государственного технического университета. Выпуск 1. Донецк: ДонГТУ, 1997.-С.193-197.
56. Дяченко О.Н. Эффективность псевдоисчерпывающего тестирования комбинационных схем // Информатика, кибернетика и вычислительная техника (ИКВТ - 97). Сборник научных трудов Донецкого государственного технического университета. Выпуск 1. Донецк: ДонГТУ, 1997.-С.188-192.
57. Дяченко О.Н., Герасимов А.Н. Компактное тестирование цифровых устройств на основе многозначной логики/ Донец. гос. техн. ун-т.- Донецк, 1997.- 16с.- Деп.. в ГНТБ Украины 12.06.97 № 356- УК97.
58. Методичні вказівки щодо виконання лабораторних робіт з курсу “Теорія інформації та кодування”. Розділ “Перешкодостійке кодування” (для студентів спеціальності 6.0915)/ Скл.: О.М.Дяченко.- Донецьк: ДонДТУ, 1998.- 14с. (на електронному носії № 2603) (0,75 др. арк.)

59. Dyachenko O.N. Memory Compact Testing with Error Burst Localization// Engineering Simulation.- 1997.- Vol.14.- pp.907-915. (0,63 др. арк.)
60. Дяченко.О.Н., Рудь.В.А. Эффективность сигнатурного анализа над полем GF(4) / Донец. гос. техн. ун–т.- Донецк , 1998.- 17с. Библиогр. 6 назв. –Рус.- Деп.в ГНТБ Украины 02.06.98 № 264-УК98.
61. Дяченко.О.Н., Горбенко.В.П. Сигнатурно-синлромный анализ над полем GF(3) / Донец. гос. техн. ун–т.- Донецк , 1998.- 20с. Библиогр. 8 назв. – Рус.- Деп.в ГНТБ Украины 02.06.98 № 262-УК98.
62. Дяченко.О.Н., Шишацкий Д.А. Комплексная оценка эффективности компактного тестирования/ Донец. гос. техн. ун–т.- Донецк , 1998.- 11с. Библиогр. 5 назв. – Рус.- Деп.в ГНТБ Украины 02.06.98 № 263-УК98.
63. Дяченко О.Н. Эффективность сигнатурного анализа в цифровых схемах с самотестированием// Электрон. моделирование, 1998.-20,№4.-С.79-87. (0,63 др.арк.)
64. Dyachenko O.N. Efficiency of Signature Analysis in Digital Circuits with Self-Testing// Engineering Simulation.- 1999.- Vol.16.- pp.503-512. (0,63 др. арк.)
65. Зінченко Ю., Дяченко О., Маркитантов В., Прокопченко В., Жебелєв О., Мірошніков А., Ритов А., Масюк А., Скабєлка В., Штукарін І., Прядко І.,Волкогон А., Резнік В. Новые hardware-технологии в ДонГТУ/ Материалы междунар. научно-техн. конф.”Новые информационные технологии в САПР и АСУ”. Киев.- 2001.-С. 12-14.
66. Zinchenko Y., Dyachenko O., Markitantov V., Prokopchenko V., Gebelev O., Miroshnikov A., Ritov I., Masyuk A., Skabelka V., Shtukarin I., Pryadko I., Volkogon A., Reznik V. New hardware-technologies at DonSTU/ Материалы междунар. научно-техн. конф.”Новые информационные технологии в САПР и АСУ”. Киев.- 2001.-С. 15-19.
67. Дяченко О.Н., Солодовников С.В. Эффективность компактного анализа над полем GF(3)// Наукові праці Донецького державного технічного університету. Серія: Інформатика, кібернетика та обчислювальна техніка, випуск 15.- Донецьк: ДонДТУ, 2000.-С.112-117.
68. Дяченко О.Н. Компактный анализ над полем GF(3)// Наукові праці Донецького національного технічного університету. Серія: Проблеми моделювання та автоматизації проектування динамічних систем (МАП-2002). Випуск: 52.- Донецьк: ДонНТУ.- 2002.-С.162-167. (0,38 др. арк.)
69. Методичні вказівки до виконання контрольної роботи “Розробка схем електричних принципіальних” з дисципліни “Основи конструювання ЕОМ” (для студентів спеціальності 7.091501 заочної форми навчання)/ Скл.: О.М.Дяченко. - Донецьк: ДонНТУ, 2004.- 34с. (на електронному носії № 4222, прот. № 11 від 01.03.04) (2,13 др. арк.).

70. Возовик К.П., Дяченко О.Н. Особенности аппаратной реализации и корректирующих возможностей кодов Рида-Соломона, исправляющих одиночные ошибки// Інформатика та комп'ютерні технології: Тез. доп. міжнародної наук.-техн. конф.- Донецьк, 15-16 грудня 2005. – С.315-316. (0,12/0,06 др. арк.).
71. Возовик К.П., Дяченко О.Н. Графический способ представления кодов Ивадари // Інформатика та комп'ютерні технології: Тез. доп. міжнародної наук.-техн. конф.- Донецьк, 15-16 грудня 2005. – С.317-318. (0,12/0,06 др. арк.).
72. Методические указания к выполнению лабораторных работ по курсу «Теория информации и кодирования» (для студентов специальности 7.091501, 7.091502)/ Сост.: О.Н.Дяченко, Ю.Е.Зинченко.-Донецк: ДонНТУ, 2006. – 93 с. - Рус. - HTML-формат (на електронному носії №5117, прот. №1 від 15.03.06.). 5,81/2,91 др.арк.
73. Волков А.О., Дяченко О.Н. Определение оптимальных параметров кодов Ивадари// Інформатика та комп'ютерні технології 2006: Тез. доп. другої міжнародної студентської наук.-техн. конф. – Донецьк, 13 груд. 2006.- С.279-280.
74. Смовзюк М.С., Дяченко О.Н. Локализация пакетов ошибок на основе кодов Рида-Соломона// Інформатика та комп'ютерні технології 2006: Тез. доп. другої міжнародної студентської наук.-техн. конф. – Донецьк, 13 груд. 2006.- С.305-306.
75. Методичні вказівки до виконання лабораторних робіт з дисципліни "Конструювання комп'ютерних систем" (для студентів спеціальності 7.091501)/ Скл.: О.М.Дяченко. - Донецьк: ДонНТУ, 2007. - 93 с. Формат eBook (на електронному носії №21, прот. №7 від 20.06.07) 10,0 др.арк.
76. Методичні вказівки до виконання лабораторних робіт з курсу «Теорія перешкодостійкого кодування» (для студентів спеціальності 7.091501)/ Скл.: О.М.Дяченко.- Донецьк: ДонНТУ, 2007. - 35с. - HTML-формат.(на електронному носії №18, прот. №7 від 20.06.07) 3,8 др.арк.
77. Методичні вказівки до виконання лабораторних робіт з курсу «Theory of antinoise coding» (для студентів спеціальності 7.091501)/ Скл.: О.М.Дяченко.- Донецьк: ДонНТУ, 2007. - 35с. - HTML-формат.(на електронному носії №19, прот. №7 від 20.06.07) 3,8 др.арк.
78. Дяченко О.Н. Графический способ представления сверточных кодов// Наукові праці Донецького національного університету. Серія "Проблеми моделювання та автоматизації проектування динамічних систем" (МАП-2006). Випуск: 5 (116) - Донецьк: ДонНТУ. - 2006. – С.169-178
79. Дяченко О.Н. Графический способ представления сверточных кодов// Наукові праці Донецького національного технічного університету. Серія

“Інформатика, кібернетика і обчислювальна техніка” (ІКОТ-2007). Випуск 8 (120) - Донецьк: ДонНТУ, 2007. – С.89-98.
80. Дяченко О.Н. Аппаратная реализация и корректирующие возможности кодов Рида-Соломона// Наукові праці Донецького національного технічного університету. Серія “Проблеми моделювання та автоматизації проектування динамічних систем” (МАП-2007). Випуск: 6 (127) - Донецьк: ДонНТУ. - 2007.– С.113-121.
81. Методические указания к выполнению лабораторных работ по курсу “Теория и проектирование помехоустойчивых кодов для защиты информации в КС” (для студентов специальности 7.091501)/ Сост.: О.Н.Дяченко -Донецк: ДонНТУ, 2008. – HTML-формат (на електронному носії № 428, прот. № 3 від 05.03.08) (4,08 др.арк.)
82. Юрьев И.В., Дяченко О.Н. Определение оптимальных параметров кодов Рида-Соломона // Інформатика та комп’ютерні технології / Матеріали V міжнародної науково-технічної конференції студентів, аспірантів та молодих науковців – 24-26 листопада 2009 р., Донецьк, ДонНТУ. – 2009. – С. 82-88.
83. Юрьев И.В., Дяченко О.Н. Влияние параметров поля Галуа и перемежения на избыточность кода Рида-Соломона // Інформаційні управляючі системи та комп’ютерний моніторинг (ІУС та КМ-2010) / Матеріали І всеукраїнської науково-технічної конференції студентів, аспірантів та молодих вчених – 19-21 травня 2010 р., Донецьк, ДонНТУ. – 2010. – С. 164-168.
84. Дяченко О.Н., Юрьев И.В. Влияние параметров кодов Рида-Соломона на избыточность кода// Наукові праці Донецького національного технічного університету. Серія “Проблеми моделювання та автоматизації проектування динамічних систем” (МАП-2010). Випуск: 8 (168) - Донецьк: ДонНТУ. - 2010.– С.49-56.
85. Юрьев И.В., Дяченко О.Н. Методика проектирования устройств компактного тестирования. // Інформатика та комп’ютерні технології / Матеріали VI міжнародної науково-технічної конференції студентів, аспірантів та молодих науковців – 23-25 листопада 2010 р., Донецьк: ДонНТУ. - Т.2 – 2010. – С. 32-36.
86. Методичні вказівки до виконання лабораторних робіт з курсу “Теорія коректуючих кодів” (для студентів спеціальності 7.091502 “Системне програмування”)/ Скл.: О.М. Дяченко, Ю.Є. Зінченко - Донецьк: ДонНТУ, 2011. – 36 с. (на електронному носії № 34, прот. №1 від 13.01.11) (2,92 др.арк.)
87. Методичні вказівки щодо організації самостійної роботи студентів при виконанні індивідуальних завдань з курсу “Теорія перешкодостійкого кодування” (для студентів спеціальності 7.091501)/ Скл.: О.М.Дяченко -

Донецьк: ДонНТУ, 2011. – 18 с. (на електронному носії № 76, прот. № 1 від 13.01.11) (1,06 др.арк.)

88. Конспект лекцій з курсу “Теорія перешкодостійкого кодування” (для студентів спеціальності 7.091502 “Комп'ютерні системи та мережі”)/ Скл.: О.М.Дяченко - Донецьк: ДонНТУ, 2011. - 85 с. (на електронному носії № 75, прот. № 1 від 13.01.11) (5,31 др.арк.)
89. Методичні вказівки щодо організації самостійної роботи студентів при виконанні індивідуальних завдань з курсу “Теорія коректуючих кодів” (для студентів спеціальності 7.091502) “Системне програмування”/ Скл.: О.М.Дяченко - Донецьк: ДонНТУ, 2011. – 38 с.(на електронному носії № 208, прот. № 2 від 21.03.11) (2,38 др.арк.)
90. Дяченко О.Н., Юрьев И.В. Компактное тестирование цифровых устройств на основе кода Рида-Соломона // Наукові праці Донецького національного технічного університету. Серія “Проблеми моделювання та автоматизації проектування” (МАП-2011). Випуск: 9 (179): - Донецьк: ДонНТУ. - 2011. – С. 36-43.
91. Зинченко Е.Ю., Дяченко О.Н. Сравнительный анализ способов укорачивания кодов Рида-Соломона // Збірка праць VII міжнародної науково-технічної конференції студентів, аспірантів та молодих науковців – 22-23 листопада 2011 р., Донецьк, ДонНТУ. – 2011. У 2-х томах, Т. 1 – С. 48-52.
92. Зинченко Е.Ю., Калашников В.И., Гайдук С., Бобровский К.В., Дяченко О.Н., Корченко А.А., Гриценко А.А., Ханаев В.В., Зинченко Т.А., Войтов Г.В., Сероштан С.Ю. Лаборатория ДонНТУ «FPGA-технологии проектирования и диагностика компьютерных систем» // Современные информационные технологии и ИТ-образование [Электронный ресурс] / Сборник научных трудов VI Международной научно-практической конференции / под ред. В.А. Сухомлина. - Москва: МГУ, 2011. - Т. 1. - С. 422-429.
93. Конвейерные устройства на FPGA/ Волошин Д.Н., Зинченко Ю.Е., Дяченко О.Н.// Інформаційні управляючі системи та комп'ютерний моніторинг (ІУСКМ-2012): ІІІ Всеукраїнська науково-технічна конференція студентів, аспірантів та молодих вчених, 16-18 квітня 2012 р., м. Донецьк / Донец. національ. техн. ун-т; редкол.: Є.О. Башков (голова) та ін. - Донецьк: ДонНТУ, 2012. - С. 581-585.