

Министерство образования Республики Беларусь  
Учреждение образования  
Белорусский государственный университет  
информатики и радиоэлектроники

Кафедра радиотехнических систем

***ИССЛЕДОВАНИЕ СВОЙСТВ КОДА  
РИДА - СОЛОМОНА***

Пособие к лабораторной работе по дисциплине  
КОДИРОВАНИЕ И ЗАЩИТА ИНФОРМАЦИИ  
для студентов специальностей

I-39 01 02 «Радиоэлектронные системы» и I-39 01 03 «Радиоинформатика»

Минск 2005

УДК 621. 391.25 (075.8)  
ББК 32.811 я 73  
И 88

Составитель:  
С. Б. Саломатин

**Исследование** свойств кода Рида – Соломона. Пособие к лаб. работе  
И 88 по дисциплине «Кодирование и защита информации» для студ. спец.  
I-39 01 02 «Радиоэлектронные системы» и I-39 01 03 «Радиоинформатика»  
/ Сост. С.Б. Саломатин. – Мн.: БГУИР, 2005. – 32 с.: ил.  
ISBN 985-444-819-3

Пособие содержит теоретические сведения, алгоритмы, программы моделирования процессов кодирования и декодирования, а также исследования структурных свойств блочных корректирующих кодов Рида – Соломона. В лабораторной работе исследуются различные методы кодирования и декодирования во временной и спектральных областях. Оценивается эффективность кодирования информации.

**УДК 621. 391. 25 (075.8)**  
**ББК 32.811 я 73**

**ISBN 985-444-819-3**

© Саломатин С. Б., составление, 2005  
© БГУИР, 2005

## **Содержание**

1. ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ
2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ
  - 2.1. Коды Рида – Соломона
  - 2.2. Алгоритм Берлекемпа – Мессе (БМ) решения ключевого уравнения
  - 2.3. Преобразование Фурье в поле Галуа
  - 2.4. Кодирование и декодирование кода РС в частотной области
  - 2.5. Исправление ошибок и стираний
3. ПРОГРАММЫ ДЛЯ ИССЛЕДОВАНИЯ СВОЙСТВ КОДА РС
4. СОДЕРЖАНИЕ И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ
  - 4.1. Содержание работы
  - 4.2. Порядок выполнения работы
5. СОДЕРЖАНИЕ ОТЧЕТА
6. КОНТРОЛЬНЫЕ ВОПРОСЫ
- ЛИТЕРАТУРА



## 1. ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

1. Изучить методы помехоустойчивого кодирования и декодирования информации с помощью циклических блочных кодов Рида – Соломона.
2. Исследовать свойства кодов Рида – Соломона (РС) во временной и спектральных областях.
3. Изучить связь корректирующей способности кодов с конечными полями и спектральными преобразованиями Фурье – Галуа.
4. Приобрести навыки построения алгоритмов и программ кодирования и декодирования информации.

## 2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

### 2.1. Коды Рида – Соломона

Коды Рида – Соломона являются важным подмножеством кодов Боуза – Чоудхури – Хоквингема (БЧХ), у которых мультипликативный порядок алфавита символов кодового слова делится на длину кода.

Коды Рида – Соломона обладают рядом замечательных особенностей:

- коды РС являются максимальными, т.е. при заданной длине  $n$  и размерности  $k$  имеют наибольшее кодовое расстояние  $d = n - k + 1$ ;
- любой набор из  $k$  позиций кодового слова является информационным, т.е. позволяет восстановить все кодовое слово;
- число кодовых слов веса  $\omega$  равно

$$A(w) = \binom{n}{w} (q-1) \sum_{i=0}^{w-d} (-1)^i \binom{w-1}{i} q^{w-d-1} \leq \binom{n}{w} (q-1)^{w-d+1},$$

где  $\binom{n}{w}$  – число сочетаний из  $n$  по  $w$ ,  $q$  – основание кода;

- класс РС-кодов используется в конструкциях альтернатных кодов, обобщенных каскадных кодов, равновесных кодов и т.д.;
- коды могут быть эффективно декодированы во временной и частотной областях.

Имеется несколько определений кодов РС. Так, код РС над полем Галуа  $GF(q)$  можно определить как код, состоящий из всех слов  $\mathbf{c} = (c_0, c_1, \dots, c_{n-1})$  длины  $n$ , для которых выполняются  $d-1$  уравнений:

$$\sum_{i=0}^{n-1} c_i g_i^r = 0 \quad c_i \in GF(q), r_0 \leq r \leq r_0 + d - 2, \quad (1)$$

где  $r_0$  и  $d$  – произвольные целые числа (не больше  $n$ );  $g_0, g_1, \dots, g_{n-1}$  – различные ненулевые элементы конечного поля, которые еще называют локатором  $i$ -й позиции кодового слова.

Число различных элементов  $g_i$  определяет максимальную длину кода РС, которая не превышает  $q - 1$ . Если длина кода равна  $n < q - 1$ , то код называется укороченным, если длина кода равна  $n = q$  (или  $q + 1$ ), то код является расширенным на один (или два) символа.

В матричном представлении система уравнений принимает вид

$$\mathbf{H}(c_0, c_1, \dots, c_{n-1}) = 0,$$

где  $\mathbf{H}$  – проверочная матрица кода размером  $(n-k) \times n$ . Принадлежность к классу максимальных кодов следует из свойств его проверочной матрицы

$$\mathbf{H} = \begin{bmatrix} g_0^{r_0} & g_1^{r_0} & \mathbf{L} & g_{n-1}^{r_0} \\ g_0^{r_{0+1}} & g_1^{r_{0+1}} & \mathbf{L} & g_{n-1}^{r_{0+1}} \\ \mathbf{M} & \mathbf{M} & \mathbf{O} & \mathbf{M} \\ g_0^{r_{0+m-1}} & g_1^{r_{0+m-1}} & \mathbf{L} & g_{n-1}^{r_{0+m-1}} \end{bmatrix} = \begin{bmatrix} 1 & 1 & \mathbf{L} & 1 \\ g_0 & g_1 & \mathbf{L} & g_{n-1} \\ \mathbf{M} & \mathbf{M} & \mathbf{O} & \mathbf{M} \\ g_0^{m-1} & g_1^{m-1} & \mathbf{L} & g_{n-1}^{m-1} \end{bmatrix} \begin{bmatrix} g_0^{r_0} & 0 & \mathbf{L} & 0 \\ 0 & g_1^{r_0} & \mathbf{L} & 0 \\ \mathbf{M} & \mathbf{M} & \mathbf{O} & \mathbf{M} \\ 0 & 0 & \mathbf{L} & g_{n-1}^{r_0} \end{bmatrix}, \quad (2)$$

где  $m = n - k$ .

Левый сомножитель в произведении матриц является частью матрицы Вандермонда. Это позволяет сделать вывод, что любые  $m$  столбцов матрицы  $\mathbf{H}$  линейно независимы. Отсюда следует, что для любого заданного набора  $\{\gamma_i\}$  и любых значений  $k$  символов кодового слова система линейных уравнений (1) над  $GF(q)$  имеет единственное решение относительно оставшихся  $m = n - k$  неизвестных.

Последовательность локаторов позиций  $\{\gamma_i\}$  в (1) произвольна и может быть задана любым удобным способом.

Рассмотрим способ, приводящий к построению циклических кодов.

Циклический РС-код длиной  $n$ , равной любому делителю  $(q - 1)$ , задается выражением (1) и последовательностью локаторов положений вида  $\gamma_i = \alpha^i$ , где  $\alpha$  – элемент порядка  $n$  в поле  $GF(q)$ ,  $a = \sqrt[n]{1}$ .

Циклический сдвиг влево на  $l$  позиций последовательности  $\gamma_i = \alpha^i$  эквивалентен умножению локаторов на  $\alpha^l$ :  $g_i a^l = a^{i+l \bmod n}$ , где  $n|(q - 1)$ .

Если некоторый вектор  $\mathbf{c} = (c_0, c_1, \dots, c_{n-1})$  удовлетворяет (1), то и после сдвига получаем

$$\sum_{i=0}^{n-1} c_i g_i^r a^{l_r} = \left( \sum_{i=0}^{n-1} c_i g_i^r \right) a^{l_r} = 0,$$

так как  $a^{l_r} \neq 0$  при любых  $l$  и  $r$ ,  $r_0 \leq r \leq r_0 + d$ .

Проверочная матрица циклического РС-кода имеет вид

$$\mathbf{H} = \left[ a^{(r_0+i)j \bmod n} \right]_{i=0, \dots, m-1, j=0, \dots, n-1}.$$

Это позволяет рассматривать умножение  $\mathbf{H}$  на вектор  $\mathbf{c}$  как вычисление значений полинома  $c(x) = \sum_i c_i x^i$  в точках  $\{a^{r_0+i}\}$ ,  $i = 0, \dots, m-1$ . Следовательно, циклический РС-код состоит из всех полиномов степени меньше  $n$  над  $GF(q)$ , корнями которых являются элементы  $\{a^{r_0+i}\}$ ,  $i = 0, \dots, m-1$ , где  $a \in GF(q)$ ,  $a^n = 1$ , и  $n|(q-1)|$ .

Это свойство циклических РС-кодов интересно тем, что позволяет определить сам код и его декодирование в понятиях преобразования над конечным полем Фурье - Галуа (ПФГ). Матрица  $\mathbf{H}$  является частью матрицы ПФГ, содержащей  $m = d-1$ , начиная со строки  $r_0$ .

*Определение синдрома.* Предположим, что на вход декодера поступает последовательность символов  $(y_0, y_1, \dots, y_{n-1})$ , где  $y_i = c_i + e_i$ ,  $\{e_i\}$  – элементы вектора ошибок,  $e_i \in GF(q)$ . Синдромом кода называется вектор  $\mathbf{s} = (s_0, s_1, \dots, s_{d-2})$ , элементы которого определяются из выражения

$$s_j = \sum_i y_i g_i^{r_0+j}, j = 0, 1, \dots, d-2. \quad (3)$$

Если все элементы синдрома равны нулю, то принятое слово считается кодовым. В противном случае принятое слово  $\mathbf{y}$  содержит ошибки. Синдром зависит только от ошибок в принятом слове.

Построим полином синдрома

$$s(x) = \sum_{j=0}^{m-1} s_j x^j = \sum_{j=0}^{m-1} x^j \sum_{i=0}^{n-1} e_i g_i^{r_0+j}, m = n - k.$$

Изменим порядок суммирования

$$s(x) = \sum_{i=0}^{n-1} e_i g_i^{r_0} \sum_{j=0}^{m-1} (x g_i)^j.$$

Используем понятие геометрической прогрессии для преобразования второй суммы:

$$\sum_j (xg_i)^j = \frac{(xg_i)^m - 1}{xg_i - 1}.$$

Тогда выражение для полинома синдрома  $s(x)$  приводится к виду

$$s(x) = x^m \sum_{i=0}^{n-1} \frac{e_i g_i^{r_0+m}}{xg_i - 1} - \sum_{i=0}^{n-1} \frac{e_i g_i^{r_0}}{xg_i - 1}. \quad (4)$$

Если предположить, что в принятом векторе имеются  $t$  ошибок, расположенных на позициях с номерами  $i_1, i_2, \dots, i_t$ , то выражение (4) может быть приведено к виду

$$s(x) \mathcal{S}(x) + x^m \Theta(x) = w(x), \quad (5)$$

где

$$\mathcal{S}(x) = \prod_{v=1}^t (xg_{i_v} - 1),$$

$$w(x) = - \sum_{v=1}^t e_{i_v} g_{i_v}^{r_0} \sum_{m=1, m \neq v}^t (xg_{i_m} - 1),$$

$$\Theta(x) = \sum_{v=1}^t e_{i_v} g_{i_v}^{r_0+m} \sum_{m=1, m \neq v}^t (xg_{i_m} - 1).$$

Полином  $\mathcal{S}(x)$  называют полиномом локаторов ошибок, так как его корни являются обратными величинами локаторов искаженных позиций. Степень  $\mathcal{S}(x)$  равна  $t$ , если число ошибок  $t < m$ .

Полином  $w(x)$  называется полиномом значений ошибок (полином ошибок). Степень  $w(x)$  всегда меньше степени  $\mathcal{S}(x)$ . Полиномы  $\mathcal{S}(x)$  и  $w(x)$  взаимно просты.

Уравнение (5) часто используется в виде модульного сравнения

$$s(x) \mathcal{S}(x) = w(x) \bmod x^m, \quad (6)$$

которое называется ключевым уравнением.

Если полиномы  $\mathcal{S}(x)$  и  $w(x)$  известны, тогда для определения локаторов ошибок достаточно найти корни полинома  $\mathcal{S}(x)$  среди обратных величин локаторов позиций кода, т.е. найти все решения уравнения  $\mathcal{S}(x^{-1}) = 0$ ,  $x \in \{\gamma_i\}$ . Если  $\mathcal{S}(x^{-1})$  имеет степень  $t$  и точно  $t$  различных локаторов кода являются его корнями, то можно считать, что в принятом векторе действительно  $t$  ошибок. Если же хотя бы один корень  $\mathcal{S}(x^{-1})$  не является локатором кодового слова, то в принятом слове более чем  $(d-1)/2$  ошибок и  $\mathcal{S}(x)$  не является правильным полиномом локаторов ошибок.

Если локаторы ошибок известны, то значения ошибок равны:

$$e_{i_v} = \frac{-w(g_{i_v}^{-1})}{g_{i_v}^{r_0} \prod_{m=1, m \neq v}^t (g_{i_v}^{-1} g_{i_m} - 1)}.$$

Введем понятие формальной производной, которая определяется через выражение



$$s'(x) = \sum_{v=1}^t g_{i_v} \prod_{m=1, m \neq v}^t (x g_{i_m} - 1).$$

Используя понятие формальной производной, выражение значения ошибки можно преобразовать к виду формулы Форни:

$$e_{i_v} = \frac{-w(g_{i_v}^{-1})g_{i_v}^{-r_0+1}}{s'(g_{i_v}^{-1})}.$$

*Алгебраический алгоритм декодирования кода РС (рис.1).*

Исходные данные: принятое слово  $y = (y_0, y_1, \dots, y_{n-1})$ ; параметры кода –  $n, k, d, m = d - 1, r_0$ , локаторы кода –  $g_0, g_1, \dots$ .

1. Вычисление синдрома  $s(x)$  принятого слова

$$s(x) = \sum_{j=0}^{m-1} s_j x^j, \quad s_j = \sum_{i=0}^{n-1} y_i g_i^{r_0+j}.$$

2. Решение ключевого уравнения  $s(x) = w(x) \bmod x^m$ . Если степень  $\deg(\sigma(x)) \geq m/2$ , то отказ и переход к 5.
3. Определение множества локаторов ошибок  $Z$ :  $z \in Z$ , если  $\sigma(g_z^{-1}) = 0$ . Если же не все корни являются локаторами позиций, то отказ и переход к 5.
4. Вычисление значений ошибок и исправление искаженных позиций  $\hat{y}_z = y_z$ ,

$$\text{если } z \notin Z, \hat{y}_z = y_z - e_z \text{ для } z \in Z, e_z = \frac{-w(g_z^{-1})g_z^{-r_0+1}}{s'(g_z^{-1})}.$$

5. Восстановление информационных символов или выдача признака отказа от декодирования.

Рассмотрим некоторые примеры кодирования кодом РС.

*Пример.* Построим код РС в конечном поле  $GF(2^m)$  характеристики 2. Каждый символ можно представить в виде набора из  $m$  бит. Выберем поле  $GF(8)$ , в котором существует примитивный элемент порядка 7. Зададим полином  $f(x) = x^3 + x + 1$  и построим расширенное поле как

$$\alpha = z, \alpha^2 = z^2, \alpha^3 = z + 1, \alpha^4 = z^2 + z, \alpha^5 = z^2 + z + 1, \alpha^6 = z^2 + 1, \alpha^7 = 1.$$

*Порождающий многочлен* кода РС задается в виде произведения

$$g(x) = (x - w)(x - w^2)(x - w^3) \dots (x - w^{2^t}), \quad (7)$$

где  $t$  – требуемое количество исправляемых ошибок,  $w$  – элемент конечного поля.

Задавая  $t = 1$  и выбирая  $w = \alpha$ , получим порождающий полином кода в виде

$$g(x) = (x - w)(x - w^2) = x^2 + a^4 x + a^3$$

или, применяя форму записи элементов поля в двоичном коде,

$$g(x) = (001)x^2 + (110)x + (011).$$

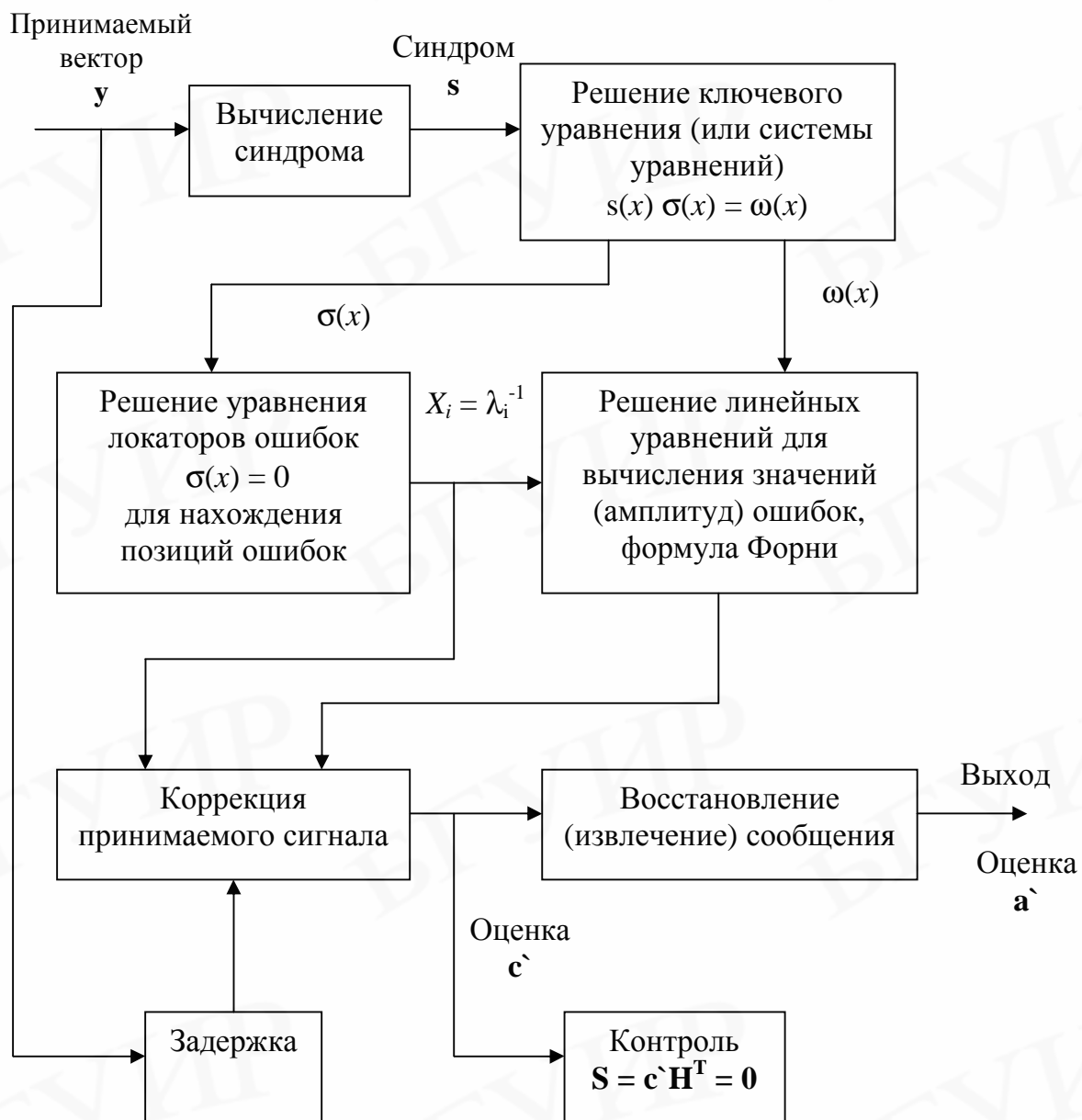


Рис. 1. Схема алгебраического декодера РС-кода во временной области

Некоторые слова несистематического кода РС (7,5) с использованием конвертирования двоичного представления символов в десятичный код приведены ниже:

$$\begin{aligned} a(x) = 0 &\rightarrow c(x) = (0, 0, 0, 0, 0, 0, 0); \\ a(x) = 1 &\rightarrow c(x) = (0, 0, 0, 0, 1, 6, 3); \\ a(x) = \alpha &\rightarrow c(x) = (0, 0, 0, 0, 2, 7, 6); \\ a(x) = \alpha^3 &\rightarrow c(x) = (0, 0, 0, 0, 3, 1, 5), \\ &\dots \quad \dots \quad \dots \quad \dots, \end{aligned}$$

весь полный список состоит из  $8^5 = 2^{15}$  кодовых слов.

*Пример.* Пусть над полем  $GF(2^4)$  с примитивным полиномом  $x^4 + x + 1$  задан РС-код с параметрами  $(n, k) = (15, 9)$ . Данный код имеет кодовое расстояние  $d = n - k + 1 = 7$ , поэтому способен исправлять до трех ошибок.

Порождающий полином имеет степень  $m = n - k = 6$  и находится по формуле (7).

Для  $v=1$  получается полином вида

$$\begin{aligned} g_1(x) &= \prod_{i=1}^6 (x + a^i) = (x + 2)(x + 3)(x + 4)(x + 5)(x + 6)(x + 7) = \\ &= x^6 + 11x^5 + 15x^4 + 5x^3 + 7x^2 + 10x + 7. \end{aligned}$$

Для  $v=0$  получается полином вида

$$\begin{aligned} g_2(x) &= \prod_{i=0}^5 (x + a^i) = (x + 1)(x + 2)(x + 3)(x + 4)(x + 5)(x + 76) = \\ &= x^6 + 10x^5 + 13x^4 + 2x^3 + 3x^2 + 5x + 7. \end{aligned}$$

Здесь  $\alpha = 2$  – примитивный элемент поля  $GF(2^4)$  и все вычисления поводятся по правилам этого поля. Предположим, что информационный вектор имеет вид

$$\mathbf{a} = (11, 13, 9, 6, 7, 15, 14, 12, 10).$$

Умножая его на порождающий полином  $g_1(x)$ , после преобразования получим несистематическое кодовое слово  $\mathbf{c} = (11, 5, 3, 15, 6, 7, 7, 5, 12, 15, 14, 3, 3, 7, 1)$ . Для того чтобы провести кодирование в систематической форме, надо вычислить остаток от деления сдвинутого на 6 позиций влево кодового слова на порождающий полином. Можно показать, что справедливо следующее соотношение:

$$\begin{aligned} (11, 13, 9, 6, 7, 15, 14, 12, 10, 0, 0, 0, 0, 0, 0) &= (1, 11, 15, 5, 7, 10, 7) \times \\ &\times (11, 15, 9, 10, 12, 10, 10, 10, 3) + (1, 2, 3, 7, 13, 9), \text{ rem} = (1, 2, 3, 7, 13, 9). \end{aligned}$$

Кодовое слово в систематическом виде имеет вид

$$\mathbf{c} = (11, 13, 9, 6, 7, 15, 14, 12, 10; 1, 2, 3, 7, 13, 9),$$

где младшие разряды расположены справа и нумеруются от нуля.

*Кодирование симметричным полиномом.*

Порождающий полином РС-кода, имеющего кодовое расстояние  $d = 2t + 1$  над полем  $GF(2^m)$ , представляется в виде

$$g(x) = \prod_{i=b}^{b+d-2} (x + \alpha^i) = (x + \alpha^b) \dots (x + \alpha^{b+d-2}), \quad (8)$$

где  $\alpha$  – примитивный элемент поля,  $b$  – некоторое положительное целое число.

Объем вычислительного процесса при кодировании можно сократить, выбрав число  $b = 2^{m-1} - t$ . В этом случае полином  $g(x) = g_0 + g_1 x + \dots + g_{2t} x^{2t}$  является симметричным, т.е. имеет симметричные коэффициенты:  $g_0 = g_{2t} = 1$ ,  $g_i = g_{2t-i}$ ,  $i = 1, 2, \dots, t-1$ .

*Пример.* Пусть задан  $(n, k) = (15, 11)$  – код РС над полем  $GF(2^4) / (x^4 + x + 1)$ . Порождающий полином кода в симметричной форме будет иметь вид

$$g(x) = (x + \alpha^6)(x + \alpha^7)(x + \alpha^8)(x + \alpha^9) = x^4 + 4x^3 + 2x^2 + 4x + 1,$$

$$t = (m/2) = 2, b = 2^{m-1} - t = 6, \alpha = 2.$$

## 2.2. Алгоритм Берлекемпа – Мессе (БМ) решения ключевого уравнения

Берлекемп предложил итерационный метод вычисления полиномов  $\Lambda(z)$  и  $\Omega(z)$ , входящих в ключевое уравнение, по изменённому полиному  $S(z)$ . Мессе трактовал алгоритм Берлекемпа как процедуру построения наиболее короткого регистра с обратными связями, порождающего  $m$  компонент синдрома  $S$ . Он же нашёл вариант алгоритма, который вычисляет только полином локаторов  $\Lambda(z)$ , коэффициенты которого удовлетворяют заданной системе уравнений [1, 2, 6].

*Задача декодера* – найти полином  $\Lambda(z)$  или  $S(z)$  минимальной степени  $v$ , удовлетворяющий системе уравнений, где  $v$  – число ошибок, причём заранее не известное. Алгоритм Берлекемпа – Мессе позволяет строить такую последовательность полиномов  $S_j(z)$ ,  $j = 1, 2, \dots, m$ , в которой каждый последующий полином удовлетворяет большему числу уравнений. Одновременно итеративно вычисляется полином величин ошибок  $\Omega(z)$ .

*Исходные данные* – это  $m$  компонент синдрома  $S$ , где обычно  $b = 1$ . Начальные условия для полиномов  $S(z), \Omega(z), C(z), D(z)$  и переменной  $L$  задаются вначале.

Алгоритм выполняется за  $m$  итераций. Если в процессе вычислений невязка  $\Delta = 0$ , то полиномы  $S(z)$  и  $\Omega(z)$  не изменяются, в противном случае осуществляется их коррекция

$$S_j(z) = S_{j-1}(z) \oplus \Delta z C_{j-1}(z);$$

$$\Omega_j(z) = \Omega_{j-1}(z) \oplus \Delta D_{j-1}(z),$$

где  $C_{j-1}(z)$  и  $D_{j-1}(z)$  – вспомогательные полиномы, взятые из предыдущего шага. На шаге  $j$  они формируются из следующих соображений. Если  $2L_{j-1} > j-1$ , то содержание регистров, хранящих эти вспомогательные полиномы, сдвигается:  $C(z) = zC(z)$ ;  $D(z) = zD(z)$ , а переменная  $L$  не изменяется. При  $2L_{j-1} \leq j-1$  указанные полиномы и переменная  $L$  трансформируются:

$$C_j(z) = \Delta^{-1} S_{j-1}(z), D_j(z) = \Delta^{-1} z \Omega_{j-1}(z), L_j = j - L_{j-1}.$$

По окончании  $m$  итераций целесообразно выполнить проверку:  $\deg S(z) = L$ ; если это равенство несправедливо, то вырабатывается сигнал  $FLAG = 1$  защитного отказа от декодирования, так как принятое слово содержит более чем  $t$  ошибок.

### 2.3. Преобразование Фурье в поле Галуа

Пусть  $\mathbf{v} = \{v_i, i = 0, \dots, n-1\}$  – вектор над  $GF(q)$ , где  $n$  делит  $q^m - 1$  при некотором  $m$ , и пусть  $\alpha$  – элемент порядка  $n$  в поле  $GF(q^m)$ . Преобразование Фурье в поле Галуа (ПФГ) вектора  $\mathbf{v}$  определяется как вектор  $\mathbf{V} = \{V_j, j = 0, 1, \dots, n-1\}$ , задаваемый равенством

$$V_j = \sum_{i=0}^{n-1} \alpha^{ij} v_i, \quad j = 0, \dots, n-1.$$

Дискретный индекс  $i$  называется временем, а  $\mathbf{v}$  – временной функцией или сигналом. Аналогично индекс  $j$  можно назвать частотой, а  $\mathbf{V}$  – частотной функцией или спектром.

В качестве длины ПФГ можно выбрать произвольный делитель числа  $(q^m - 1)$ , но наиболее важную роль играют примитивные длины  $n = (q^m - 1)$ . В последнем случае  $\alpha$  является примитивным элементом поля  $GF(q^m)$ .

Если  $m$  – наименьшее целое, такое, что  $n$  делит  $(q^m - 1)$ , то над полем  $GF(q)$  существует преобразование Фурье длины  $n$  и компоненты этого преобразования лежат в поле  $GF(q^m)$ . Временная функция  $\mathbf{v}$  принимает значения в поле  $GF(q)$ , а её спектр  $\mathbf{V}$  лежит в расширении поля  $GF(q^m)$ . В приложениях, связанных с контролем ошибок, все связанные с декодированием действия осуществляются в большом поле  $GF(q^m)$ , однако начинать надо с вектора входных данных (на входе канала), т.е. в малом поле  $GF(q)$ .

Над полем  $GF(q)$  характеристики  $p$  вектор и его спектр связаны соотношениями

$$V_j = \sum_{i=0}^{n-1} v_i \alpha^{ij}, \quad v_i = (n^{-1}) \sum_{j=0}^{n-1} V_j \alpha^{-ij}.$$

В матричном виде эти соотношения запишутся как

$$\mathbf{V} = [\alpha^{ij}] \mathbf{v}^T, \mathbf{v} = [\alpha^{ij}]^{-1} \mathbf{V}^T,$$

где  $[\alpha^{ij}]$ ,  $[\alpha^{ij}]^{-1}$  – прямая и обратная матрицы преобразования Фурье,  $T$  – оператор транспонирования.

Одним из сильнейших свойств ПФГ является *свойство свертки*. Предположим, что  $e_i = f_i g_i, i = 0, \dots, n-1$ . Тогда

$$E_j = \left(n^{-1}\right) \sum_{k=0}^{n-1} F_{((j-k))} G_k, \quad j = 0, \dots, n-1,$$

где двойные скобки означают, что индексы вычисляются в арифметике по модулю  $n$ .

Другой замечательной особенностью ПФГ является *свойство сдвига*.

Если  $\{v_i\} \leftrightarrow \{V_j\}$  является парой ПФГ, то парами ПФГ являются также

$$\{a^i v_i\} \leftrightarrow \{V_{((j+1))}\} \text{ и } \{v_{((i-1))}\} \leftrightarrow \{a^i V_j\}.$$

Иногда вектор  $\mathbf{v}$  задается многочленом  $v(x)$ . С помощью ПФГ многочлен

$$v(x) = v_{n-1}x^{n-1} + \dots + v_1x + v_0$$

может быть преобразован в многочлен

$$V(x) = V_{n-1}x^{n-1} + \dots + V_1x + V_0,$$

который называется спектральным многочленом или многочленом, ассоциированным с многочленом  $v(x)$ .

Справедливы следующие свойства спектра, связанные с корнями многочлена:

- 1) элемент  $\alpha^i$  является корнем многочлена  $v(x)$  тогда и только тогда, когда  $j$ -я частотная компонента  $V_j$  равна нулю;
- 2) элемент  $\alpha^i$  является корнем многочлена  $V(x)$  тогда и только тогда, когда  $i$ -я временная компонента  $v_i$  равна нулю.

Таким образом, если в одном случае говорят о корнях многочлена в конечном поле, а в другом – о равных нулю спектральных компонентах, то в действительности говорят об одном и том же.

Свойства ПФГ позволяют дать описание циклических кодов. Каждое слово  $c(x)$  задается многочленом степени  $(n-1)$ . В несистематическом виде оно может быть записано как  $c(x) = g(x) a(x)$ , где  $a(x)$  – информационный многочлен степени  $(k-1)$ . Во временной области это дает циклическую свертку

$$c_i = \sum_{l=0}^{n-1} g((i-l)) a_l.$$

Следовательно, в частотной области операция кодирования может быть записана в виде произведения

$$C_j = G_j A_j.$$

Любой удовлетворяющий этому равенству спектр задает в частотной области кодовое слово при условии, что во временной области все компоненты являются  $GF(q)$ -значными. В силу произвольности информационного спектра единственная существенная роль компонент  $G_j$  состоит в том, чтобы определить частоты, в которых стоят нулевые компоненты  $C_j$  спектра кодового слова.

*Спектральное определение циклического кода.* Циклическим кодом называется множество таких слов над  $GF(q)$ , у которых все спектральные компоненты, принадлежащие заданному множеству так называемых проверочных частот  $j_1, \dots, j_{n-k}$ , равны нулю.

Хотя каждое слово циклического кода является вектором над  $GF(q)$ , спектр кодового слова является вектором  $GF(q^m)$ . Следовательно, циклический код может быть определен как множество  $GF(q)$ -значных обратных преобразований Фурье множества всех спектральных векторов, компоненты которых в заданном множестве частот равны нулю. Нельзя выбирать произвольный спектральный вектор, у которого стоят нули в заданном множестве частот; обратные преобразования некоторых из таких векторов могут иметь компоненты, не принадлежащие полю  $GF(q)$ . Для того чтобы кодовое слово принадлежало полю  $GF(q)$ , выбирать нужно только спектр, который удовлетворяет следующим условиям сопряженности.

Пусть  $\mathbf{V}$  есть  $n$ -мерный вектор с компонентами из  $GF(q^m)$ , где  $n$  делит  $(q^m - 1)$ . Тогда обратное преобразование Фурье  $\mathbf{v}$  является вектором с компонентами из  $GF(q)$  тогда и только тогда, когда выполняются следующие равенства:

$$V_j^q = V_{(qj)}, \quad j = 0, 1, \dots, n-1.$$

Чтобы применить это свойство, разобьем числа по  $\text{mod } n$  на подмножества элементов циклотомических классов:

$$C_j = \{j, jq, \dots, jq^{m_j-1}\},$$

где  $m_j$  – наименьшее целое положительное число, удовлетворяющее равенству  $jq^{m_j} = j \text{ mod } n$ .

Например, если  $q = 2$ ,  $n = 7$ , то элементы циклотомических классов имеют вид  $C_0 = \{0\}$ ,  $C_1 = \{1, 2, 4\}$ ,  $C_2 = \{3, 6, 5\}$ .

Циклотомический класс выделяет в спектре множество частот. Назовем эти частоты хордой. Тогда, если временной сигнал принимает значения в поле  $GF(q)$ , то значение спектра в одной из частот хорды определяет значения спектра при всех частотах этой хорды.

*Пример.* Вычислим прямое и обратное ПФГ над полем  $GF(17)$ . Зададим примитивный элемент поля  $\alpha = 3$ . Элементы поля образуют множество

$$[1, 3, 9, 10, 13, 5, 15, 11, 16, 14, 8, 7, 4, 12, 2, 6].$$

Множество обратных элементов поля равно

$$\{\alpha^{-1}\} = \{6, 2, 12, 4, 7, 8, 14, 16, 11, 15, 5, 13, 10, 9, 3, 1\}.$$

Матрица прямого преобразования размером  $16 \times 16$  имеет вид

$$V = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 3 & 9 & 10 & 13 & 5 & 15 & 11 & 16 & 14 & 8 & 7 & 4 & 12 & 2 & 6 \\ 1 & 9 & 13 & 15 & 16 & 8 & 4 & 2 & 1 & 9 & 13 & 15 & 16 & 8 & 4 & 2 \\ 1 & 10 & 15 & 14 & 4 & 6 & 9 & 5 & 16 & 7 & 2 & 3 & 13 & 11 & 8 & 12 \\ 1 & 13 & 16 & 4 & 1 & 13 & 16 & 4 & 1 & 13 & 16 & 4 & 1 & 14 & 16 & 4 \\ 1 & 5 & 8 & 6 & 13 & 14 & 2 & 10 & 16 & 12 & 9 & 11 & 4 & 3 & 15 & 7 \\ 1 & 15 & 4 & 9 & 16 & 2 & 13 & 8 & 1 & 15 & 4 & 9 & 16 & 2 & 13 & 8 \\ 1 & 11 & 2 & 5 & 4 & 10 & 8 & 3 & 16 & 6 & 15 & 12 & 13 & 7 & 9 & 14 \\ 1 & 16 & 1 & 16 & 1 & 16 & 1 & 16 & 1 & 16 & 1 & 16 & 1 & 16 & 1 & 16 \\ 1 & 14 & 9 & 7 & 13 & 12 & 15 & 6 & 16 & 3 & 8 & 10 & 4 & 5 & 2 & 11 \\ 1 & 8 & 13 & 2 & 16 & 9 & 4 & 15 & 1 & 8 & 13 & 2 & 16 & 9 & 4 & 15 \\ 1 & 7 & 15 & 3 & 4 & 11 & 9 & 12 & 16 & 10 & 2 & 14 & 13 & 6 & 8 & 5 \\ 1 & 4 & 16 & 13 & 1 & 4 & 16 & 13 & 1 & 4 & 16 & 13 & 1 & 4 & 16 & 13 \\ 1 & 12 & 8 & 11 & 13 & 3 & 2 & 7 & 16 & 5 & 9 & 6 & 4 & 14 & 15 & 10 \\ 1 & 2 & 4 & 8 & 16 & 15 & 13 & 9 & 1 & 2 & 4 & 8 & 16 & 15 & 13 & 9 \\ 1 & 6 & 2 & 12 & 4 & 7 & 8 & 14 & 16 & 11 & 15 & 5 & 13 & 10 & 9 & 3 \end{bmatrix}.$$

Зададим вектор сообщения в виде  $\mathbf{a} = (1, 1, 1, 3, 4, 1, 9, 0, 0, 0, 0, 0, 0, 0, 0, 0)$ .

Спектр Фурье–Галуа сообщения равен

$$\mathbf{A} = \mathbf{a} \mathbf{V} = (14, 13, 11, 9, 9, 13, 16, 11, 7, 9, 8, 3, 1, 16, 11, 3).$$

Функция спектра ПФГ показана на рис. 2.

Обратное преобразование ФГ вычислим по формуле  $a_i = \frac{1}{16} \sum_{j=0}^{15} v_j a^{-ij} \bmod 17$ .

Проведя необходимые вычисления, нетрудно убедиться, что исходное сообщение восстанавливается полностью. Так,  $a_1 = (1 / 16) [14 \cdot 1 + 13 \cdot 6 + 11 \cdot 2 + 9 \cdot 12 + 9 \cdot 4 + 13 \cdot 7 + 16 \cdot 8 + 11 \cdot 14 + 7 \cdot 16 + 9 \cdot 11 + 8 \cdot 15 + 3 \cdot 5 + 1 \cdot 13 + 16 \cdot 10 + 11 \cdot 9 + 3 \cdot 3] = 1 \bmod 17$ .

#### 2.4. Кодирование и декодирование кода РС в частотной области

Кодирование в частотной области РС-кодами над полем  $GF(q = 2^\mu)$  состоит в следующем (рис. 3). Информационное сообщение  $\mathbf{a}$  из  $k = n - m = q - 1$  символов представляется в виде старших коэффициентов спектра  $\mathbf{F}_G$  Фурье–Галуа (ФГ), а остальные  $m$  символов этого спектра полагаются равными нулю.

Кодовое слово  $\mathbf{c}$  в несистематическом виде получается с помощью обратного  $n$ -точечного преобразования ФГ спектра  $\mathbf{F}_G$ .



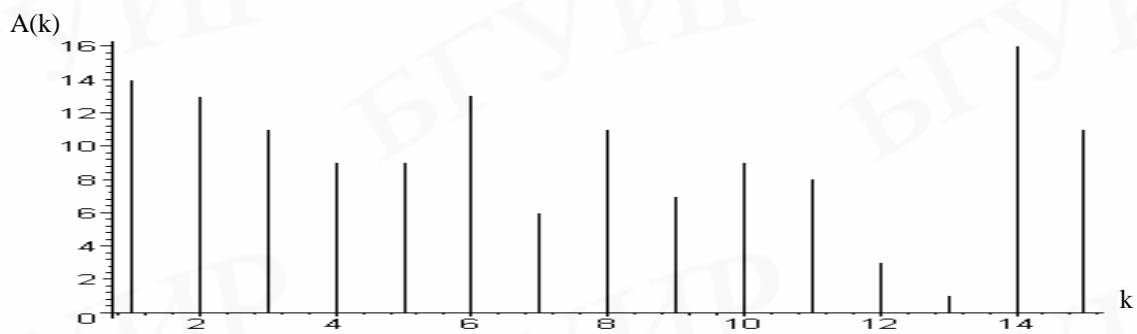


Рис. 2. Спектр кодового слова

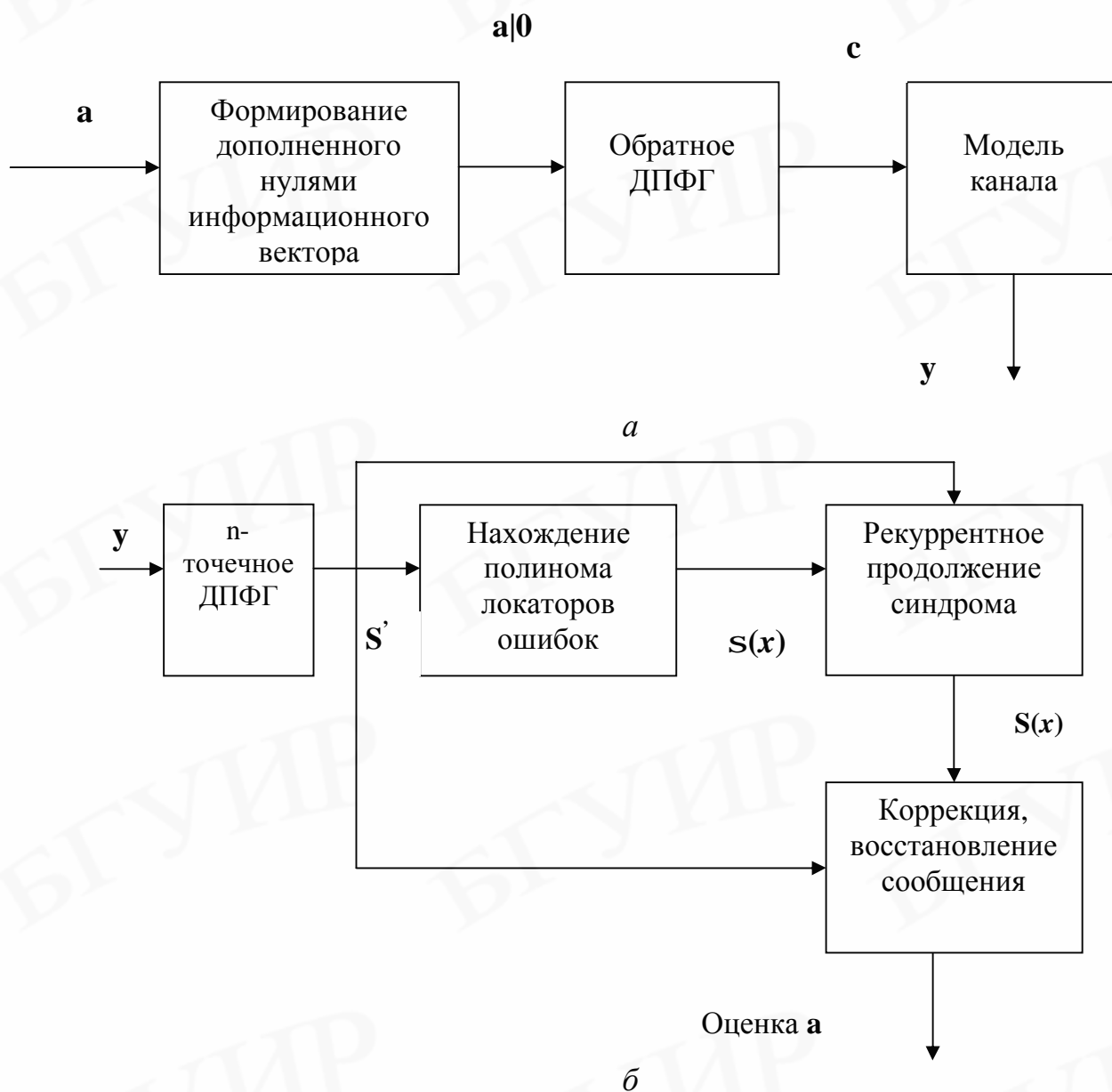


Рис. 3. Схема спектрального кодера-декодера: *а* – кодирование, *б* – декодирование

*Пример.* Пусть над полем  $GF(q = 2^4)$  необходимо закодировать сообщение кодом РС с параметрами  $n = 15$ ,  $k = 11$ ,  $d = 3$ . Элементы поля имеют вид  $1, \alpha, \alpha^2, \alpha^3, 1 + \alpha, \alpha + \alpha^2, \alpha^2 + \alpha^3, 1 + \alpha + \alpha^3, 1 + \alpha^2, \alpha + \alpha^3, 1 + \alpha + \alpha^2, \alpha^3 + \alpha^2 + \alpha, 1 + \alpha + \alpha^2 + \alpha^3, 1 + \alpha^2 + \alpha^3, 1 + \alpha^3$ .

Сообщение задано в форме  $\mathbf{a} = (a_{11}, a_{10}, \dots, a_1) = (13, 7, 15, 5, 9, 1, 0, 0, 0, 0, 0) = [a^{13}, a^7, a^{15}, a^5, a^9, \bar{0}, 0, 0, 0, 0, 0]$ .

Сформируем спектр  $\mathbf{V} = (\mathbf{a}; \mathbf{0}) = (13, 7, 15, 5, 9, 1, 0, 0, 0, 0, 0, \dots, 0)$ , где  $\mathbf{0}$  – нулевой вектор длины  $m = n - k$ . Воспользовавшись обратным ДПФГ, найдем искомое кодовое слово

$$\mathbf{c} = \mathbf{v} = \text{ДПФГ}^{-1}\{\mathbf{V}\} = (14, 0, 1, 10, 0, 5, 0, 10, 2, 0, 2, 14, 15, 0, 3).$$

#### *Алгоритм декодирования в спектральной области*

Основные этапы спектрального декодирования:

- 1) дискретное преобразование Фурье–Галуа (ДПФГ);
- 2) нахождение полинома локаторов;
- 3) рекуррентное продолжение синдрома;
- 4) коррекция и извлечение сообщения.

На первом этапе принятое слово  $\mathbf{y} = \mathbf{c} + \mathbf{e}$  подвергается  $n$  – точечному ДПФГ. Полученный вектор  $\mathbf{Y} = (Y_0, Y_1, \dots, Y_{n-1})$  соответствует спектру для суммы кодового слова и ошибки  $(\mathbf{c} + \mathbf{e})\mathbf{W}_n = \mathbf{C} + \mathbf{E}$ , где  $\mathbf{W}_n$  – матрица ДПФГ. Так как  $m$  составляющих вектора  $\mathbf{V}$  являются нулевыми, то соответствующие им  $m$  компонент вектора  $\mathbf{s} = (s_0 = X_k, s_1 = X_{k+1}, \dots, s_{m-1} = X_{n-1})$  характеризуют конфигурацию ошибок  $\mathbf{e}$ . Другими словами, эти  $m$  компонент аналогичны синдрому во временной области, причем, если все они равны нулю, то считается, что в принятом слове нет ошибок.

Второй этап проще по сравнению с временным декодированием, так как отпадает необходимость в полном решении ключевого уравнения, а достаточно найти лишь полином локаторов  $\sigma(x)$ .

Третий этап сводится к вычислению спектра  $\mathbf{E} = \{E_i\}$  ошибок. Для этого осуществляется рекуррентное продолжение синдрома по формуле

$$E_{(n-1-l-j)} \psi_\lambda = E_{(n-1-l-j)} \psi_{(l-1)} + \dots + E_{(n-2-j)} \psi_1 + E_{(n-1-j)} \psi_0,$$

где  $\psi_i$  – коэффициент при  $x^i$  полинома локаторов ошибок  $s(x)$ ,  $j = 0, 1, \dots, (n - \lambda)$ ,  $l$  – степень полинома  $s(x)$ .

При этом  $m$  старших коэффициентов спектра  $\mathbf{E}$  совпадают с соответствующими компонентами синдрома, характеризующими ошибку  $\mathbf{e}$ :

$$E_{(n-1)} = s_{m-1}, E_{(n-2)} = s_{m-2}, \dots, E_{(k=n-m)} = s_0.$$

Коррекция на четвертом этапе состоит в сложении векторов  $\mathbf{Y}$  и  $\mathbf{E}$ , а также выделении сообщения из полученной после отбрасывания нулевого вектора  $\mathbf{0}$  длины  $m$  из суммы.

*Пример.* Предположим, что кодовое слово имеет вид

$$\mathbf{c} = (14, 0, 1, 10, 0, 5, 0, 10, 2, 0, 2, 14, 15, 0, 3).$$

После искажения в канале принят вектор  $\mathbf{y} = (0, 3, 1, 10, 0, 5, 0, 10, 2, 0, 2, 14, 15, 0, 3)$ . Вектор ошибок равен  $\mathbf{e} = (e_0 = 14, e_1 = 3)$ .

После выполнения 15-точечного ДПФГ определяем спектр

$$\mathbf{Y} = \mathbf{y} \mathbf{W}_{15} = (6, 15, 11, 4, 3, 11, 4, 11, 10, 5, 2; 0, 3, 7, 13) = (Y_0, Y_1, \dots, Y_{10}; Y_{11}, \dots, Y_{14}).$$

Так как четыре правых коэффициента спектра отличны от нуля, то в принятом слове имеются ошибки.

Используя алгоритм БМ, находим полином локаторов ошибок, который равен

$$s(x) = x^2 + \alpha^3 x + \alpha^{14}.$$

Определим спектр ошибки как продолжение синдрома. Для рассматриваемого случая имеем  $n = 15$ ,  $\lambda = 2$ ,  $\psi_0 = 15$ ,  $\psi_1 = 4$ ,  $E_{14} = s_3 = 13$ ,  $E_{13} = s_2 = 7$ ,  $E_{12} = s_1 = 3$ ,  $E_{11} = s_0 = 0$ .

Убедимся, что при  $j = 0$  и  $j = 1$  рекурсивные соотношения выполняются:

$$E_{12} = E_{13} \psi_1 + E_{14} \psi_0 = \alpha^7 \alpha^3 + \alpha^{13} \alpha^{14} = \alpha^3 = 3, \quad E_{11} = E_{12} \psi_1 + E_{13} \psi_0 = 3 \cdot 3 + 7 \cdot 14 = 0.$$

Для  $j = 2$  находим

$$E_{10} = E_{11} \psi_1 + E_{12} \psi_0 = 0 \cdot 3 + 3 \cdot 14 = 2.$$

Аналогичным образом получаем:  $E_9 = 5$ ,  $E_8 = 10$ ,  $E_7 = 11$ ,  $E_6 = 4$ ,  $E_5 = 6$ ,  $E_4 = 1$ ,  $E_3 = 8$ ,  $E_2 = 12$ ,  $E_1 = 9$ ,  $E_0 = 15$ .

После сложения вектора ошибки с полученным сообщением и исключения четырех правых нулевых коэффициентов извлекаем верное сообщение.

Спектральный метод декодирования значительно проще декодирования во временной области. Однако он имеет недостаток. Сбои в блоке ДПФГ, искажающие любые несиндромные символы с номерами  $i = 0, 1, \dots, k-1$ , не могут быть выявлены.

## 2.5. Исправление ошибок и стираний

В режиме стирания фиксируются не сами оценки принятых символов, а их местоположение и им присваивается статус стёртого символа. Суть исправления и стирания состоит в том, что после декодирования можно провести восстановление стёртых символов, используя алгоритмы интерполяции. Если при декодировании использовать  $l$  стёртых символов, тогда два кода будут отличаться друг от друга по меньшей мере на  $(d - l)$  позиций, где  $d$  – кодовое расстояние. Тогда в дополнение к стиранию можно будет исправлять  $t_m = \lfloor (d - l - 1) / 2 \rfloor$  ошибок, где  $\lfloor x \rfloor$  – это целая часть числа  $x$ . Код может исправлять все комбинации из  $n$  ошибок и  $l$  стираний в канале, для которого  $2n + l < d$ .

Для восстановления одного стертого символа необходим только один проверочный символ – для восстановления значения, т.к. позиция стирания принимающей стороне известна. Например, для поля Галуа  $GF(256)$  РС-код (94, 88)-битовых символов имеет  $n = 94$ ,  $k = 88$  и может исправить до трех ошибок ( $t = 3$ ) и восстановить до шести стертых символов.

*Алгоритм исправления стираний.* Предположим, что выполнено  $f$  стираний при приеме кодового слова, в котором имеется  $v$  ошибок.

Обозначим локаторы ошибок как  $X_1 = a^{i_1}, X_2 = a^{i_2}, \dots, X_n = a^{i_n}$ , а стираний – как  $Y_{c,1} = a^{j_1}, Y_{c,2} = a^{j_2}, \dots, Y_{c,f} = a^{j_f}$ . Декодирование ведется в следующем порядке.

1. Вычисляется полином локаторов стираний:

$$\Gamma(x) = \prod_{l=1}^f (1 - Y_{c,l} \cdot x) .$$

2. В декодируемом векторе заменяют символы с координатами стираний на нулевые символы. Для нового вектора находится полином синдрома стираний  $s(x)$ .

3. Определяется модифицированный полином синдрома

$$SE(x) = (\Gamma(x)[1 + s(x)] - 1) \bmod x^{2t+1} .$$

4. Вычисляется полином локаторов ошибок  $S(x)$ , используя для этого алгоритм БМ и значения модифицированного полинома  $SE_i, i = f + 1, \dots, 2t$ .

5. Определяются корни уравнения  $S(x) = 0$  и координаты ошибок.

6. Составляется ключевое уравнение

$$w(x) = S(x)[1 + SE(x)] \bmod x^{2t+1}$$

и определяется полином локаторов ошибок-стираний  $Y(x) = S(x)\Gamma(x)$ .

7. Оцениваются значения ошибок и стираний. Значения ошибок вычисляют по формуле

$$Q_{i_k} = \frac{-X_k w(X_k^{-1})}{y'(X_k^{-1})} , \text{ где } y' - \text{ формальная производная.}$$

Значения стираний вычисляют по формуле  $F_{i_k} = \frac{-Y_k w(Y_k^{-1})}{y'(Y_k^{-1})}$ .

*Пример.* Декодируется код РС (7, 3), построенный над полем  $GF(8)$ .

Принимаемый вектор равен

$$\begin{aligned} y(x) &= \alpha^4 x^6 + \alpha^5 x^5 + \alpha^2 x^4 + x^3 + \alpha^6 x^2 + \alpha^5 x + \alpha^6 = \\ &= y_6 x^6 + y_5 x^5 + y_4 x^4 + y_3 x^3 + y_2 x^2 + y_1 x + y_0 . \end{aligned}$$

Приемник выдал стирания на позициях  $j_1 = 1, j_2 = 6$ , что позволяет записать полином стираний как  $er(x) = er_1 x + er_2 x^2$ . Алгоритм декодирования:

1.  $\Gamma(x) = (1 - \alpha x)(1 - \alpha^6 x) = 1 + \alpha^5 x + x^2$ ;

2. Заменяем в  $y(x)$  символы на 1-й и 6-й позициях нулями:

$$y_c(x) = \alpha^5 x^5 + \alpha^2 x^4 + x^3 + \alpha^6 x^2 + \alpha^6, \text{ после чего вычисляем синдром}$$

$$S_1 = y_c(\alpha) = \alpha; S_2 = y_c(\alpha^2) = \alpha; S_3 = y_c(\alpha^3) = \alpha; S_4 = y_c(\alpha^4) = \alpha^3;$$

$$s(x) = \alpha x + \alpha x^2 + \alpha x^3 + \alpha^3 x^4.$$

3. Модифицируем полином синдрома

$$\begin{aligned} SE(x) &= [(1 + \alpha^5 x + x^2)(\alpha x + \alpha x^2 + \alpha x^3 + \alpha^3 x^4) - 1] \bmod x^5 = \\ &= \alpha^6 x + \alpha^4 x^2 + \alpha^6 x^3 + \alpha^2 x^4. \end{aligned}$$

4. Применяя алгоритм БМ, получим выражение для полинома локаторов ошибок  $S(x) = 1 + \alpha^3 x$ . Локатор ошибки  $X_1 = \alpha^3$ . Ошибка расположена на третьей позиции.

5. Ключевое уравнение равно

$$\begin{aligned} w(x) &= S(x)[1 + SE(x)] \bmod x^5 = \\ &= (1 + \alpha^3 x)(1 + \alpha^6 x + \alpha^4 x^2 + \alpha^6 x^3 + \alpha^2 x^4) \bmod x^5 = 1 + \alpha^4 x + \alpha x^2 + \alpha^2 x^3. \end{aligned}$$

Полином  $y(x) = (1 + \alpha^2 x + \alpha^3 x^2 + \alpha^3 x^3)$ .

Формальная производная  $y'(x) = (\alpha^2 + \alpha^3 x^2)$ .

6. Вычисляем значения ошибок и стираний:

$$Q_3 = \frac{a^3 w(a^4)}{y'(a^4)} = a^6, \quad F_1 = \frac{a w(a^6)}{y'(a^6)} = a^5, \quad F_6 = \frac{a^6 w(a)}{y'(a)} = a^4.$$

*Спектральное представление процесса исправления ошибок и стираний.*

Дополнительно введем следующие обозначения:

-  $\Gamma i = \{i_0, i_1, \dots, i_{r-1}\}$  – множество индексов позиций  $r$ -стираний;

-  $M = \{j_0, j_1, \dots, j_{n-r-1}\}$  – множество индексов элементов декодируемого вектора, не затронутых процессом стирания;

-  $\Gamma = (g_0, g_1, \dots, g_{r-1})$  – вектор стертых символов;

-  $\mathbf{b} = (b_0, b_1, \dots, b_{n-r-1}) = (c_{j_0}, c_{j_1}, \dots, c_{j_{n-r-1}})$  – кодовый вектор, в котором исключены символы, соответствующие стираемым позициям.

Определим в проверочной матрице  $\mathbf{H}$  подматрицу  $\mathbf{H}_x$ , элементы которой имеют индексы, соответствующие позициям стираний:

$$\mathbf{H}_x = \begin{bmatrix} 1 & 1 & \mathbf{L} & 1 \\ x_0 & x_1 & \mathbf{L} & x_{r-1} \\ \mathbf{L} & \mathbf{L} & \mathbf{O} & \mathbf{L} \\ x_0^{r-1} & x_1^{r-1} & \mathbf{L} & x_{r-1}^{r-1} \end{bmatrix},$$

где  $x_m = a_{i_m}$ ,  $i_m \in \Gamma i$ ,  $m = 0, 1, \dots, r-1$ .

Символы  $(x_0, x_1, \dots, x_{r-1})$  имеют смысл локаторов стираний.

Матрица  $\mathbf{W}$ , соответствующая остальным, не стертым символам, имеет вид

$$\mathbf{W} = \begin{bmatrix} 1 & 1 & \mathbf{L} & 1 \\ w_0 & w_1 & \mathbf{L} & w_{n-r-1} \\ \mathbf{L} & \mathbf{L} & \mathbf{O} & \mathbf{L} \\ w_0^{r-1} & w_1^{r-1} & \mathbf{L} & w_{n-r-1}^{r-1} \end{bmatrix},$$

где  $w_m = a_{j_m}$ ,  $j_m \in \mathbf{M}$ ,  $m = 0, 1, \dots, n - r - 1$ .

Модифицированный синдром можно определить из выражения

$$SE^T = (s_0, s_1, \dots, s_{r-1})^T = \mathbf{Wb}^T.$$

Если в принятом векторе заменить стерты символы нулями

$$\mathbf{y} = (y_0, y_1, \dots, y_{n-1}), \quad y_i = \begin{cases} 0, & i \in \Gamma \\ c_i, & i \in \mathbf{M} \end{cases},$$

то синдром может быть вычислен по формуле

$$s_i = \sum_{j=0}^{n-1} y_j a^{ij}, \quad i = 0, 1, \dots, n - k - 1.$$

Для каждого кодового слова можно составить уравнение

$$\mathbf{H}_x \mathbf{g}^T + \mathbf{Wb}^T = 0.$$

Отсюда вектор стертых символов может быть определен как

$$\mathbf{g}^T = -\mathbf{H}_x^{-1} \mathbf{Wb}^T = -\mathbf{H}_x^{-1} SE^T.$$

Применение той или иной формы представления и декодирования диктуется техническими требованиями на декодер.

### 3. ПРОГРАММЫ ДЛЯ ИССЛЕДОВАНИЯ СВОЙСТВ КОДА РС

**VMP := proc(v::vector, A::array, p::integer)** - вычисление произведения вектора  $\mathbf{v}$  на матрицу  $\mathbf{A}$ ,  $\mathbf{vA}$  по модулю  $p \pmod{p}$ ;

**GenMat:=proc(G::array, p::integer)** - формирование генераторной матрицы ступенчатой формы  $G \pmod{p}$  для кода, определяемого строкой матрицы  $G$ ;

**DualGenMat:=proc(G::array, p::integer)** - вычисление генераторной матрицы дуального кода, определяемого строкой матрицы  $G \pmod{p}$  (формой генераторного полинома);

**ToPoly** - преобразование вектор - полином с коэффициентами, равными элементам вектора

> **ToPoly:=proc(w::array, p::integer, var::symbol)**

```

local i,n,poly;
n:=vectdim(w);
poly:=0;
for i to n do
  poly:=poly+(Normal(w[i]) mod p)*var^(i-1);
end do;
RETURN(sort(poly))
end:

```

**ToVector** – конвертирование полинома в вектор длины n, n должно быть больше или равно величине  $\text{degree}(\text{poly}, \text{var}) + 1$

> **ToVector:=proc(poly::polynom,var::symbol,n::integer,p::integer)**

```

local i,v;
v:=vector(n);
for i to n-1 do
  v[i+1]:=Normal(coeff(poly,var^i)) mod p;
end do;
v[1]:=Normal(subs(var=0,poly)) mod p;
RETURN(eval(v))
end:

```

**IrredPoly** – выбор неприводимого полинома степени r над полем  $F(p)$  путем факторизации полинома  $x^{(p^r-1)}-1 \bmod p$

> **IrredPoly:=proc(r::integer,p::integer)**

```

local fp,i,irr;
fp:=Factor(x^(p^r-1)-1) mod p;
for i to nops(fp) do
  if degree(op(i,fp)) = r then
    irr:=op(i,fp);
    break
  end if;
end do;
RETURN(irr)
end:

```

**MinPoly** – вычисление минимального полинома элемента  $\text{beta}^s$ , где  $\text{beta} = \text{RootOf}(h)$  – примитивный элемент поля  $\text{GF}(p^r)$ , где  $r = \text{deg}(h)$ . Полином h должен быть неприводимым над полем  $\text{GF}(p)$  и выполнено неравенство  $1 \leq s \leq 2^r-2$

> **MinPoly:=proc(s::integer,h::polynom,p::integer,var::symbol)**

```

local CC,n,beta,t,mp;
beta:=RootOf(h);
n:=p^degree(h)-1;
if evalb((1 <= s) and (s <= n-1)) then
  CC:=CyclotomicCoset(s,n,p);
  mp:=1;
  for t in CC do
    mp:=mp*(var-beta^t);
  end do;
  mp:=expand(mp);
  mp:=Normal(mp) mod p;
  RETURN(sort(mp))
else
  lprint("Error -- s not in correct range 1 <= s <= 2^degree(h)-2")
end if
end:

```

**RSGenPoly** – нахождение генераторного полинома кода РС. Оператор "irred" – должен быть неприводимым полиномом степени  $r$  над полем  $F_p$ , корни которого являются примитивными элементами для поля  $GF(p^r)$ . Корнями генераторного полинома являются элементы  $\beta^{(m+1)}, \dots, \beta^{(m+\delta-1)}$

```

> RSGenPoly:=proc(p::integer,r::integer,delta::integer,m::integer,
  h::polynom,var::symbol)
local i,beta,gp;
beta:=RootOf(h);
gp:=1;
for i from m+1 to m+delta-1 do
  gp := gp*(var - beta^i);
end do;
RETURN(sort(collect(Normal(expand(gp)) mod p,var))) end:

```

**RSEncode** – полиномиальная форма кодирования информации кодом РС. Оператор "delta" определяет минимальное кодовое расстояние;  $\delta = \text{degree}(\text{gen}) + 1$ . Длина информационного слова должна быть равна  $p^r - \delta$ . Переменная "var" должна совпадать с символикой переменной генераторного полинома

```

RSEncode:=proc(info::array,gen::polynom,var::symbol,irred::polynom,p::integer)
local r,delta,s,f,cp,i;
r:=degree(irred);
delta:=degree(gen)+1;

```



```

s:=vectdim(info);
if s = p^r-delta then
  f:=0;
  for i to s do
    f:=f+info[i]*var^(p^r-i-1);
  end do;
  cp:= f - Rem(f,gen,var) mod p;
  RETURN(ToVector(cp,var,p^r-1,p))
else
  lprint('Error -- expecting p^r - delta information symbols')
end if; end:

```

**RSBMSynPol** – вычисление синдрома ошибки принятого вектора. Результат представляется в виде полинома переменной  $x$

```

> RSBMSynPol:=proc(received::array,delta::integer,m::integer,irred::polynom,
  p::integer)
  local r,n,beta,wp,syndromes,j,sp;
  r:=degree(irred);
  n:=p^r-1;
  if evalb((vectdim(received) = n) and (modp(delta,2) = 1)) then
    beta:=RootOf(irred);
    wp:=ToPoly(received,p,x);
    syndromes:=array(m+1..m+delta-1);
    for j from m+1 to m+delta - 1 do
      syndromes[j]:=Normal(subs(x=beta^j,wp)) mod p;
    end do;
    sp:=1;
    for j to delta - 1 do
      sp:=sp+syndromes[m+j]*x^j;
    end do;
    RETURN(sp)
  else
    lp('Error -- received word has incorrect length, or delta even') end if; end:

```

**RSBMErrLoc** – вычисление полинома локаторов ошибок по алгоритму Берлекемпа-Мессис. Минимальное кодовое расстояние "delta" определяется как нечетное число. "irred " – неприводимый полином степени  $r$ , соответствующий степени расширенного поля  $GF(p^r)$ . Синдром "syndromepoly" должен иметь форму полинома относительно переменной  $x$

```

> RSBMErrLoc:=proc(syndromepoly::polynom,delta::integer,irred::polynom,

```

```

    p::integer,verbose::boolean)    local q,pp,z,DD,cc,cd,i;
q:=array(-1..delta-1);
pp:=array(-1..delta-1);
z:=array(-1..delta-1);
DD:=array(-1..delta-1);
q[-1]:=syndromepoly;
q[0]:=simplify((syndromepoly-1)/x);
pp[-1]:=x^delta;
pp[0]:=x^(delta-1);
DD[-1]:=-1; DD[0]:=0; z[0]:=-1;
for i to delta-1 do
    cc:=Normal(subs(x=0,q[i-1])) mod p;
    if cc = 0 then
        q[i]:=simplify(q[i-1]/x);
        pp[i]:=simplify(pp[i-1]/x);
        DD[i]:=2+DD[i-1];
        z[i]:=z[i-1];
    else
        cd:= Normal( subs(x=0,q[z[i-1]]) ) mod p;
        q[i]:=Normal(simplify((q[i-1]-(cc/cd)*q[z[i-1]])/x)) mod p;
        pp[i]:=Normal(simplify((pp[i-1]-(cc/cd)*pp[z[i-1]])/x)) mod p;
        DD[i]:=2+min(DD[i-1],DD[z[i-1]]);
        if DD[i-1] >= DD[z[i-1]] then
            z[i]:=i-1;
        else
            z[i]:=z[i-1];
        end if;
    end if;
    if verbose then
        print("step",i,"");
        print(ToPoly(ToVector(pp[i],x,degree(pp[i])+1,p),p,x),
            ToPoly(ToVector(q[i],x,degree(q[i])+1,p),p,x));
    end if;    end do;
RETURN(ToPoly(ToVector(pp[delta-1],x,degree(pp[delta-1])+1,p),p,x))    end:
RSBMErrVal – вычисление местоположения ошибок и значений ошибок РС-кода
(p^r,delta);
> RSBMErrVal:=proc(errloc::polynom,synpoly::polynom,m::integer,delta::integer,
    irred::polynom,p::integer)
    local r,rr,e,errrev,roots,i,l,beta,errorvalues,rtinv,j;

```

```

r:=degree(irred);
beta:=RootOf(irred);
e:=degree(errloc);
errorvalues:=vector([seq(0,i=0..p^r-2)]);
roots:=array(0..p^r-2,[seq(0,i=0..p^r-2)]);
errrev:=Normal(simplify(x^e*subs(x=1/x,errloc))) mod p;
j:=0;
for i from 0 to p^r-2 do
  if Normal(subs(x=beta^i,errrev)) mod p = 0 then    roots[i]:=1;
    j:=j+1;    end if;  end do;
if j <> e then
  RETURN(FAIL)
else
  rr:=Rem(errloc*sypoly,x^delta,x) mod p;
  for i from 0 to p^r-2 do
    if roots[i] = 1 then
      rtinv:=Normal(1/beta^i) mod p;
      errorvalues[i+1]:=Normal(beta^(i*(1-m))*
        subs(x=rtinv,rr)/subs(x=rtinv,diff(errloc,x))) mod p;
    end if;  end do;
  RETURN(eval(errorvalues))  end if;  end:

```

**RSBMDecode** – алгоритм общего декодирования кода РС с восстановлением кодового слова;

```

RSBMDecode:=proc(rec::array,p::integer,irred::polynom,delta::integer,m::integer)
  local sp,el,errorvector;
  sp:=RSBMSynPol(rec,delta,m,irred,p);
  el:=RSBMErrLoc(sp,delta,irred,p,false);
  errorvector:=convert(RSBMErrVal(el,sp,m,delta,irred,p),vector);
  RETURN(map(y->Normal(y) mod p, matadd(rec,errorvector)));  end:

```

**CHerr** – оператор модели канала передачи информации. Моделирует генератор ошибок, который формирует случайным образом в заданном конечном поле вектор ошибок определенного веса, а также сумматор, который складывает в поле GF(p) элементы вектора кодового слова и вектора ошибок.

**Dganal** – преобразование данных, представленных в виде вектора элементов поля вида  $b^j$ , вектор десятичных чисел {j}; используется для получения функций, иллюстрирующих процессы кодирования и декодирования.

**MFGT** – формирование матрицы прямого ПФГ;

**OMFGT** – формирование матрицы обратного ПФГ;

**fx** – оператор-функция прямого ПФГ.

```
> fx:= x-> simplify(add(fin[i]*x^(i-1), i=1..k)) mod p;
      ofx – оператор-функция обратного ПФГ
> ofx:= x-> simplify((1/16)*add(fin[i]*x^(-(i-1)), i=1..k)) mod p;
```

*Пример.*

Код РС ( $n = 15, d = 5$ ) над полем  $GF(2^4 = 8)$ ,  $p = 2, r = 4$

1. Определим неприводимый полином

```
> IrredPoly(4,2);
```

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

2. Вычислим генераторный полином кода РС. Корень – примитивный элемент поля "beta" является корнем неприводимого полинома.

```
> alias(beta=RootOf(x^4+x+1));
```

```
> g:=RSGenPoly(2,4,5,0,x^4+x+1,x);
```

$$g := x^4 + (b^3 + b^2 + 1)x^3 + (b^3 + b^2)x^2 + b^3x + (b^2 + b + 1).$$

3. Зададим информационный вектор a, используя для этого:

– генератор случайного полинома в конечном поле

```
> h:=Randpoly(10,x,beta) mod 2;
```

$$\begin{aligned} h := & (b^3 + b + 1)x^{10} + (b^3 + b^2 + 1)x^9 + (b^3 + b^2 + b + 1)x^8 + \\ & + (b^3 + b^2 + b)x^7 + (b^3 + b^2 + 1)x^6 + (b^2 + 1)x^5 + (b^3 + b^2 + b)x^4 + \\ & + (b^3 + b + 1)x^3 + (b^3 + b^2 + b)x + (b^2 + b) ; \end{aligned}$$

– определив информационный вектор как множество

```
> a:=array(1..11):
```

– конвертировав случайный полином в вектор заданного размера

```
a:=ToVector(h,x,11,2);
```

$$\begin{aligned} a := & [b^2 + b, b^3 + b^2 + b, 0, b^3 + b + 1, b^3 + b^2 + b, b^2 + 1, b^3 + b^2 + b, \\ & b^3 + b^2 + b + 1, b^3 + b^2 + 1, b^3 + b + 1] \end{aligned}$$

4. Вычислим кодовое слово  $c(x)$ , представленное в полиномиальной форме

```
> c:=RSEncode(a,g,x,x^4+x+1,2);
```

$$\begin{aligned} c := & [b^2 + b + 1, b^3 + b^2 + 1, b^2 + 1, b^3 + b^2, b^3 + b + 1, b^3 + b^2 + 1, \\ & b^3 + b^2 + b + 1, b^3 + b^2 + b, b^3 + b^2 + 1, b^2 + 1, b^3 + b^2 + b, b^3 + b + 1, \\ & 0, b^3 + b^2 + b, b^2 + b]. \end{aligned}$$

5. Проведем *моделирование канала передачи информации*. Воспользуемся подпрограммой **GHerr**, которая формирует случайным образом вектор ошибки "Error", вычисляет вес Хэмминга этого вектора и складывает элементы информационного вектора с элементами вектора ошибки, формируя тем самым принимаемый вектор y:

```
> y:=GHerr(c,1,x^4+x+1,15,2);
```

$$Error = [0, 0, 0, 0, 0, 0, 0, 0, 0, b^2 + b^3, 0, 0, 0, 0, b + 1]$$

$$y := [b^2 + b + 1, b^3 + b^2 + 1, b^2 + 1, b^3 + b^2, b^3 + b + 1, b^3 + b^2 + 1, b^3 + b^2 + b + 1, b^3 + b^2 + b, b^3 + b^2 + 1, b^3 + 1, b^3 + b^2 + b, b^3 + b + 1, 0, b^3 + b^2 + b, b^2 + 1]$$

*Декодирование принятого сигнала*

6. Вычислим полином синдрома ошибок  $s(x)$

> **s:=RSBMSynPol(y,5,0,x^4+x+1,2);**

$$s := 1 + (1 + b^3)x + (b^3 + b^2 + b)x^2 + (b^3 + b)x^3 + (b^3 + b^2 + b)x^4$$

7. Вычислим полином локаторов ошибок  $q(x)$

> **q:=RSBMErrLoc(s,5,x^4+x+1,2,i=1);**

$$q := (b^2 + 1)x^2 + (b^3 + b^2 + b)x^2 + (b^3 + b)x^3 + (b^3 + b^2 + b)x^4$$

8. Вычислим оценку вектора ошибок  $E$

> **ER:=RSBMErrVal(q,s,0,5,x^4+x+1,2);**

$$ER := [0, 0, 0, 0, 0, 0, 0, 0, 0, b^3 + b^2, 0, 0, 0, 0, b + 1]$$

*Вывод.* Вектор ошибки и оценка вектора ошибки совпадают.

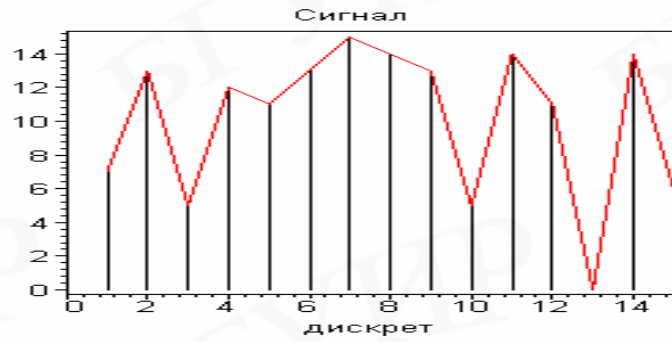
9. Выполним полное декодирование и восстановление кодового слова в виде его оценки  $vc$ . Информационная часть результата занимает  $(p^r - d)$  правых позиций вектора и имеет инверсную нумерацию индекса положения относительно положений символов в кодовом слове п. 4

> **vc:=RSBMDecode(y,2,x^4+x+1,5,0);**

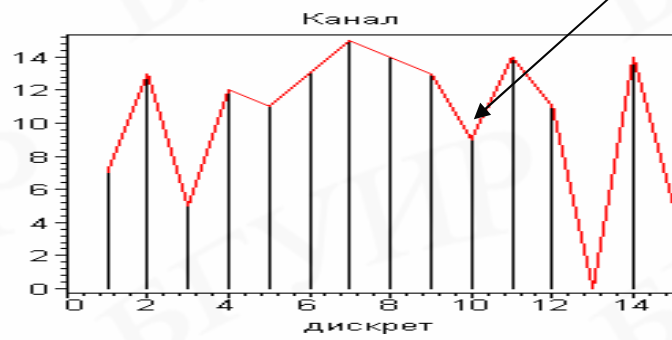
$$vc := [b^2 + b + 1, b^3 + b^2 + 1, b^2 + 1, b^3 + b^2, b^3 + b + 1, b^3 + b^2 + 1, b^3 + b^2 + b + 1, b^3 + b^2 + b, b^3 + b^2 + 1, b^2 + 1, b^3 + b^2 + b, b^3 + b + 1, 0, b^3 + b^2 + b, b^3 + b^2]$$

*Вывод.* Код РС исправил двойную ошибку, произошедшую в канале передачи информации.

На рис.4, 5 приведены функции, иллюстрирующие процессы кодирования и декодирования с положительным и отрицательным исходами.



*a*

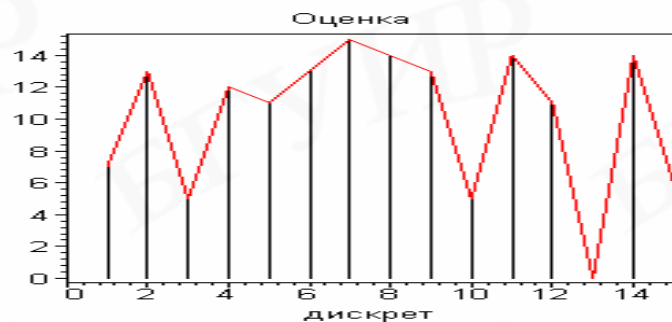


*б*

Ошибки



*в*



*г*

Рис. 4. Аналоговые функции, иллюстрирующие процесс кодирования и декодирования кода РС без ошибки: *a* – сигнальная функция; *б* – функция принимаемого сигнала; *в* – функция синдрома; *г* – функция результата декодирования (неверная оценка)

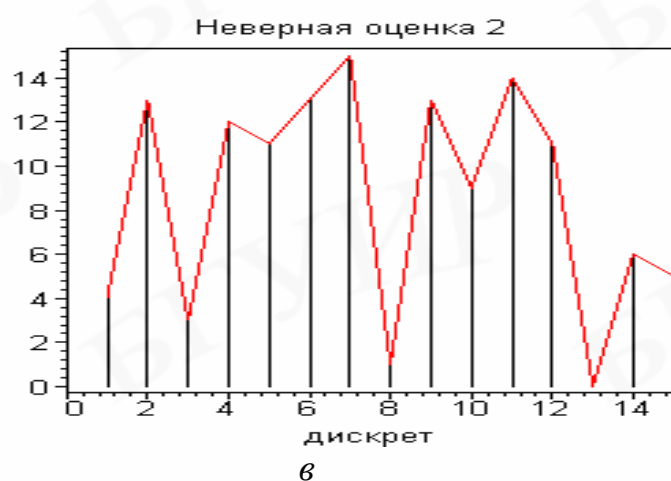
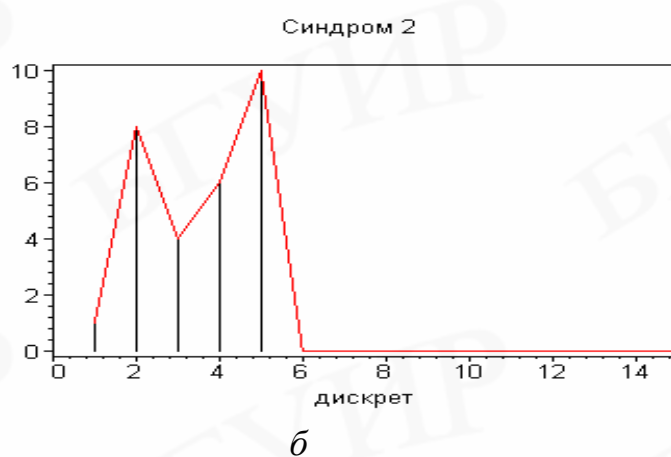


Рис. 5. Графики, иллюстрирующие процесс кодирования и декодирования кода РС с ошибкой: *a* – функция принимаемого сигнала; *б* – функция синдрома; *в* – функция оценки результата декодирования

## **4. СОДЕРЖАНИЕ И ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ**

### **4.1. Содержание работы**

1. Изучить принцип построения, свойства и алгоритмы декодирования РС-кодов.
2. Изучить правила пользования программного обеспечения к лабораторной работе и программного пакета Maple.
3. Провести моделирование алгоритмов формирования полей Галуа.
4. Провести моделирование алгоритмов преобразования Фурье–Галуа.
5. Провести моделирование с помощью программного обеспечения алгоритмов кодирования и декодирования РС-кодов во временной и спектральной областях.
6. Провести анализ спектральных и весовых характеристик кодов.
7. Оценить достоверность передачи информации с помощью РС-кодов и алгоритмическую сложность их кодирования и декодирования.

### **4.2. Порядок выполнения работы**

Составить и отладить моделирующие алгоритмы программного обеспечения.

#### *4.2.1. Моделирование алгоритмов формирования конечных полей .*

1. Получить от преподавателя исходные данные полей, построить конструкции полей, необходимые для кодирования и декодирования сообщений.

#### *4.2.2. Моделирование алгоритмов преобразования Фурье –Галуа .*

1. Получить от преподавателя исходные данные сообщений, построить конструкции прямых и обратных ПФГ.
2. Вычислить спектры сообщений и провести восстановление данных из спектральных компонент.

#### *4.2.3. Моделирование аддитивного канала передачи информации.*

1. Построить модель канала передачи информации, задав количество ошибок.
2. Сформировать вектор ошибки и принимаемый сигнал.

#### *4.2.4. Исследование спектральных характеристик кода.*

1. Вычислить спектры с помощью обратного ПФГ нескольких информационных сообщений с различными добавленными нулевыми сегментами для вычисления синдрома и наложением вектора шума.

2. Вычислить синдромы принимаемого сигнала двумя методами: для временной области и спектральной областей, используя алгоритм Берлекемпа–Мессис, прямое ПФГ и рекуррентный алгоритм восстановления синдрома – по совокупности его компонент. Провести сравнение алгоритмов вычисления и полученных результатов.

#### *4.2.5. Моделирование алгоритма алгебраического декодирования во временной области (АДВО) .*

1. Используя данные, полученные в предыдущих пунктах, провести моделирование алгоритма декодирования РС-кода алгебраическим методом во



временной области для различного числа задаваемых ошибок. Использовать алгоритмы Берлекемпа–Мессии, Форни.

2. Оценить помехоустойчивость кодирования информации кодом РС с заданными параметрами.

#### *4.2.6. Моделирование алгоритма декодирования в частотной области .*

1. Используя данные, полученные в предыдущих пунктах, провести моделирование алгоритма декодирования РС-кода спектральным методом в частотной области для различного числа задаваемых ошибок. Использовать алгоритмы ПФГ, Берлекемпа–Мессии для вычисления полинома локаторов ошибок, рекурсивный алгоритм восстановления вектора ошибок.

### ***Расчетная часть***

#### *4.2.7. Построить модель декодера, исправляющего ошибки и стирания кода РС.*

1. Получить от преподавателя исходные данные для декодирования, построить алгоритм кодирования и декодирования и провести расчет этапов исправления ошибок и стираний кодера и декодера.

## **5. СОДЕРЖАНИЕ ОТЧЕТА**

1. Формулировка цели работы.
2. Схемы алгоритмов формирования и декодирования кодов.
3. Результаты моделирования.
4. Анализ свойств исследуемых кодов и их влияние на эффективность кодирования и декодирования.
5. Оценка эффективности передачи информации с помощью РС-кодов.
6. Выводы.

## **6. КОНТРОЛЬНЫЕ ВОПРОСЫ**

1. Пояснить сущность метода формирования РС-кодов с помощью элементов конечного поля.
2. Что такое конечные поля Галуа и как они влияют на способность кода исправлять ошибки ?
3. Как выбрать порождающие полиномы для формирования РС-кода, обеспечивающего коррекцию  $t$  ошибок и заданную скорость передачи информации.

4. Как связаны функция синдрома ошибки и спектральная характеристика принимаемого сигнала, закодированного кодом РС?
5. Поясните принципы работы алгоритмов вычисления синдромов в частотной и временной областях.
6. Как оценить значение ошибки в спектральном методе декодирования?
7. Как изменится сложность декодирования при использовании симметричных порождающих полиномов?

## **ЛИТЕРАТУРА**

1. Касами Т., Токура Н., Ивадари Ё., Инагаки Я. Теория кодирования: Пер. с яп. – М.: Мир, 1978.
2. Блейхут Р. Теория и практика кодов, контролирующих ошибки: Пер. с англ. – М.: Мир, 1986.
3. Саломатин С.Б. Кодирование информации в радиоэлектронных системах. Учеб. пособие по курсу «Кодирование и защита информации» для студ. спец. «Радиоэлектронные системы» и «Радиоинформатика». – Мн.: БГУИР, 2005.
4. Саломатин С.Б. Исследование свойств кодов БЧХ. Метод. пособие по курсу «Кодирование и защита информации» для студ. спец. «Радиоэлектронные системы» и «Радиоинформатика». – Мн.: БГУИР, 2004.
5. Мак Вильямс Ф. Дж., Слоэн Н. Дж. Теория кодов, исправляющих ошибки. – М.: Связь, 1979.
6. Скляр Б. Цифровая связь. Теоретические основы и практическое применение.: Пер. с англ. – М.: Изд. дом «Вильямс», 2003.

Учебное издание

## **ИССЛЕДОВАНИЕ СВОЙСТВ КОДА РИДА – СОЛОМОНА**

Пособие к лабораторной работе по дисциплине  
**КОДИРОВАНИЕ И ЗАЩИТА ИНФОРМАЦИИ**  
для студентов специальностей I-39 01 02  
«Радиоэлектронные системы» и I-39 01 03 «Радиоинформатика»

Составитель:  
**Саломатин Сергей Борисович**

Редактор Т.Н. Крюкова  
Корректор Т.П. Андрейченко

Подписано в печать 04.08.2005.  
Гарнитура «Таймс».  
Уч.-изд. л. 1,4.

Формат 60x84 1/16.  
Печать ризографическая.  
Тираж 150 экз.

Бумага офсетная.  
Усл. печ. л. 2,09.  
Заказ 129.

---

Издатель и полиграфическое исполнение: Учреждение образования  
«Белорусский государственный университет информатики и радиоэлектроники»  
Лицензия на осуществление издательской деятельности №02330/0056964 от 01.04.2004.  
Лицензия на осуществление полиграфической деятельности №02330/0131518 от 30.04.2004.  
220013, Минск, П. Бровки, 6