ФЕДЕРАЛЬНОЕ АГЕНТСТВО СВЯЗИ

Федеральное государственное образовательное бюджетное учреждение высшего профессионального образования «САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ТЕЛЕКОММУНИКАЦИЙ им. проф. М. А. БОНЧ-БРУЕВИЧА»

С. С. Владимиров

МАТЕМАТИЧЕСКИЕ ОСНОВЫ ТЕОРИИ ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ

Курс лекций

СПб ГУТ)))

Санкт-Петербург
2014

8. Коды Рида-Соломона

Коды Рида-Соломона (коды РС) представляют из себя недвоичные циклические коды Боуза-Чоудхури-Хоквингема, символы которых представляют собой элементы поля Галуа GF(q), где $q = 2^m$ — порядок поля, а m — степень поля Галуа [34].

Коды РС (n,k) определены на всех m-битовых символах при всех n и k. для которых верно неравенство:

$$0 < k < n \leq 2^m - 1,$$

где k — число информационных символов, подлежащих кодированию, а n — число кодовых символов в кодируемом блоке. Для большинства (n,k) кодов РС

$$(n,k) = (q-1, q-1-2t), \tag{111}$$

где t — количество ошибок, исправляемых кодом, а $n-k = 2t = r$ — число контрольных символов [35].

Если $n < q-1$, то код РС называют укороченным, если $n = q$ или $n = q+1$, то код РС называется расширенным на 1 или 2 символа, соответственно [36].

Для задания циклических кодов РС используется порождающий многочлен $g(x)$ степени $d_{\min} - 1 = n-k = r$ вида [34, 21]:

$$g(x) = \prod_{j=b}^{b+2t-1} (x + \varepsilon^j), \tag{112}$$

где b — целое число. Обычно $b = 0$ или $b = 1$.

Коды РС, у которых $b = 1$, то есть, образующий полином равен

$$g(x) = (x+\varepsilon)(x+\varepsilon^2)\ldots(x+\varepsilon^{n-k}), \tag{113}$$

называют *кодами РС в узком смысле* [37].

На основе порождающего многочлена можно построить проверочный многочлен $h(x)$ степени k, удовлетворяющий условию [36]:

$$h(x)g(x) \equiv 0 \pmod{x^n + 1}.$$

При этом все кодовые слова циклического кода кратны $g(x)$, а именно

$$s(x)h(x) \equiv 0 (\text{mod } x^n + 1) \;\text{ и }\; f(x) \equiv 0 \pmod{g(x)} \tag{114}$$

Код РС обладает наибольшим минимальным кодовым расстоянием, возможным для линейного кода с одинаковыми n и k. Для недвоичных кодов

расстояние между двумя кодовыми словами определяется по аналогии с расстоянием Хемминга как число символов, которыми отличаются последовательности. Для кодов РС минимальное расстояние определяется как [38]:

$$d_{\min} = n - k + 1. \tag{115}$$

Максимальное количество ошибок, исправляемых кодом РС можно выразить следующим образом [35]:

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor = \left\lfloor \frac{n-k}{2} \right\rfloor. \tag{116}$$

Здесь $\lfloor x \rfloor$ означает наибольшее целое, не превышающее x.

Как следует из уравнений (111) и (116), коды РС, исправляющие t символьных ошибок, требуют $2t$ контрольных символов, т. е. декодер использует $n-k$ избыточных символов, количество которых вдвое превышает количество исправляемых ошибок [35].

Способность кода РС к коррекции стираний выражается следующим образом [35]:

$$\rho \leq d_{\min} - 1 = n - k, \tag{117}$$

т. е. код РС способен исправить любой набор $n-k$ или менее стираний в кодовом слове.

Возможность одновременной коррекции ошибок и стираний можно выразить как требование [35]:

$$2t' + \vartheta < d_{\min} < n - k, \tag{118}$$

где t' — число символьных ошибок, которые можно исправить, а ϑ — количество исправяемых символьных стираний.

В качестве примера рассмотрим циклический код РС $(n,k) = (7,3)$ над полем $\mathrm{GF}(2^3)$ с примитивным элементом ε, образующий полином которого имеет вид $p(x) = x^3 + x + 1$.

Согласно формуле (115) минимальное кодовое расстояние для данного кода равно:

$$d_{\min} = n - k + 1 = 5. \tag{119}$$

Количество гарантированно исправляемых кодом ошибок рассчитаем по формуле (116):

$$t = \left\lfloor \frac{n-k}{2} \right\rfloor = 2. \tag{120}$$

По формуле (112) построим порождающий многочлен кода РС, приняв $b = 1$:

$$g_{(7,3),b=1}(x) = \prod_{j=1}^{4}(x+\varepsilon^j) \begin{aligned} &= (x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^3)(x+\varepsilon^4) = \\ &= x^4 + \varepsilon^3 x^3 + x^2 + \varepsilon x + \varepsilon^3. \end{aligned} \tag{121}$$

Таким образом, был получен порождающий многочлен кода, который можно использовать для кодирования информационных комбинаций.

Если взять $b = 0$, то получится другой полином

$$g_{(7,3),b=0}(x) = \prod_{j=0}^{3}(x+\varepsilon^j) \begin{aligned} &= (x+1)(x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^3) = \\ &= x^4 + \varepsilon^2 x^3 + \varepsilon^5 x^2 + \varepsilon^5 x + \varepsilon^6, \end{aligned} \tag{122}$$

Который также может быть использован для построения кода Рида-Соломона.

8.1. Кодирование кодов РС

Кодирование с помощью кода РС может быть реализовано двумя способами: систематическим и несистематическим.

Представим исходное информационное слово как информационный полином $u(x)$ степени $k-1$, а кодовое слово кода РС в виде полинома $v(x)$ степени $n-1$.

В случае несистематического кодирования кодовое слово находится из соотношения

$$v(x) = u(x)g(x)$$

где $g(x)$ — порождающий многочлен кода РС.

Данный алгоритм реализуется кодирующим устройством, которое показано на рисунке 8.1 в общем виде.

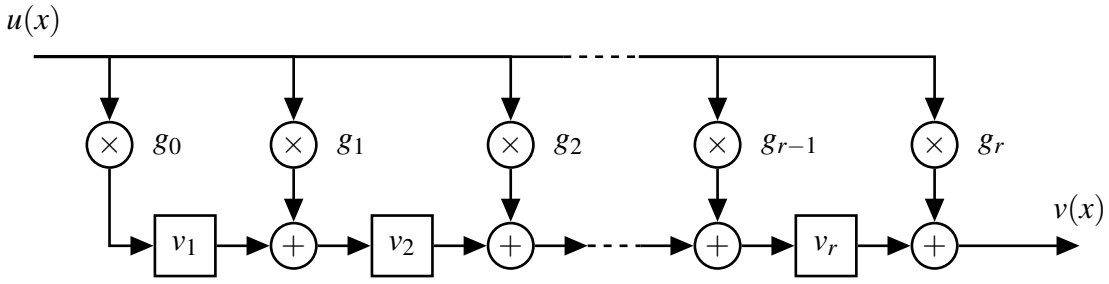

Рис. 8.1. Блок-схема кодирующего устройства несистематического кода РС

Данное устройство представляет из себя обычную схему перемножения двух многочленов.

При этом результирующая кодовая комбинация не содержит в явном виде исходных информационных элементов.

При систематическом кодировании используется следующий алгоритм [35]:

1. Осуществляется сдвиг информационного полинома $u(x)$ в крайние старшие k разрядов кодового слова путем умножения полинома $u(x)$ на x^{n-k}.
2. Полученный полином $x^{n-k}u(x)$ делится на порождающий многочлен $g(x)$ для получения остатка от деления $r(x)$.
3. Искомое кодовое слово $v(x)$ определяется как $v(x) = x^{n-k}u(x) + r(x)$.

На рисунке 8.2 приведена схема, реализующая вышеприведенный алгоритм кодирования [39].

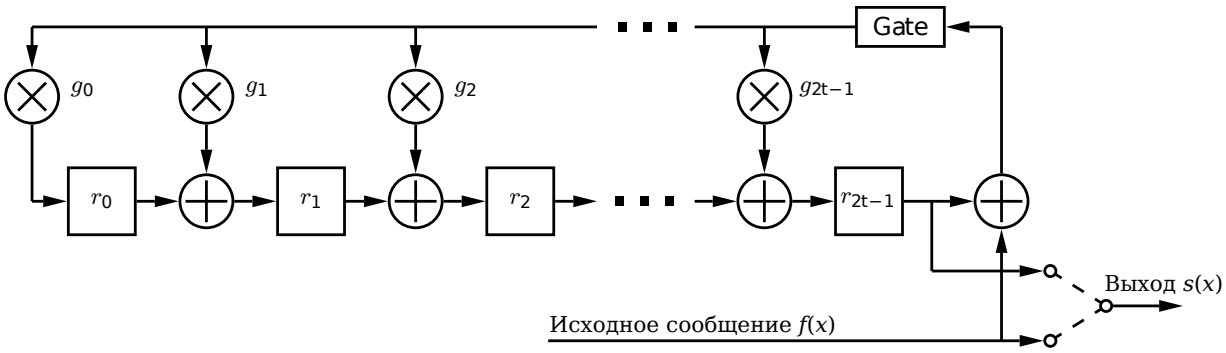

Рис. 8.2. Блок-схема кодирующего устройства систематического кода РС

При использовании данного метода, результирующее кодовое слово содержит в явном виде исходные информационные элементы.

Также существует алгоритм кодирования, не требующий использования порождающего полинома [34]. При использовании данного метода, искомое кодовое слово $v(x)$ определяется как:

$$\mathbf{v} = \{u(1)u(\varepsilon)u(\varepsilon^2)\ldots u(\varepsilon^{q-2})\},$$

где ε — примитивный элемент поля Галуа $\mathrm{GF}(q)$. Можно видеть, что этот метод кодирования также является несистематическим.

Приведем примеры кодирования информационного полинома $u(x) = \varepsilon^4 + \varepsilon^3 x^2$ всеми тремя способами, для кода РС $(7,3)$, образованного посредством полинома $g(x) = g_{(7,3),b=0}(x)$ (122).

Несистематическое кодирование.

$$\begin{aligned} v(x) = u(x)g(x) &= (\varepsilon^4 + \varepsilon^3 x^2)(x^4 + \varepsilon^2 x^3 + \varepsilon^5 x^2 + \varepsilon^5 x + \varepsilon^6) = \\ &= \varepsilon^3 + \varepsilon x + \varepsilon^4 x^2 + \varepsilon^2 x^3 + \varepsilon^5 x^4 + \varepsilon^5 x^5 + \varepsilon^3 x^6. \end{aligned}$$

Систематическое кодирование.

$$\begin{aligned} v(x) &= x^4 u(x) + \mathrm{mod}\left(\frac{x^4 u(x)}{g(x)}\right) = \\ &= (\varepsilon^4 x^4 + \varepsilon^3 x^6) + \mathrm{mod}\left(\frac{\varepsilon^4 x^4 + \varepsilon^3 x^6}{x^4 + \varepsilon^2 x^3 + \varepsilon^5 x^2 + \varepsilon^5 x + \varepsilon^6}\right) = \\ &= \varepsilon^2 + \varepsilon^2 x + \varepsilon x^2 + \varepsilon^4 x^3 + \varepsilon^4 x^4 + \varepsilon^3 x^6, \end{aligned}$$

где mod — функция, обозначающая остаток от деления.

Кодирование без использования порождающего полинома.

$$\begin{aligned} \mathbf{v} &= \{u(1)u(\varepsilon)u(\varepsilon^2)u(\varepsilon^3)u(\varepsilon^4)u(\varepsilon^5)u(\varepsilon^6)\} = \\ &= \{\varepsilon^6\ 1\ \varepsilon^5\ \varepsilon\ 0\ \varepsilon^3\ \varepsilon^2\}, \\ v(x) &= \varepsilon^6 + x + \varepsilon^5 x^2 + \varepsilon x^3 + \varepsilon^3 x^5 + \varepsilon^2 x^6. \end{aligned}$$

8.2. Алгебраический метод декодирования

Алгебраический метод декодирования кодов Рида-Соломона аналогичен процедуре декодирования двоичных кодов БЧХ. Отиличем является то, что элементами кода являются элементы соответствующего поля Галуа, и помимо позиций ошибок необходимо определить их значения.

1. Вычисление синдрома $S(x)$ принятого слова.
2. Решение ключевого уравнения $S(x)\sigma(x) \equiv \omega(x) \pmod{x^d}$. Если

$$\deg \sigma(x) \geq r/2,$$

   то отказ и переход к пункту (5).

   $\sigma(x)$ — многочлен локаторов ошибок (его корни являются обратными величинами локаторов искаженных позиций); его степень равна t.

   $\omega(x)$ — многочлен значений ошибок; его степень всегда меньше степени $\sigma(x)$.
3. Определение множества локаторов ошибок по процедуре Ченя. Если число локаторов ошибок не совпадает со степенью $\sigma(x)$, то отказ и переход к пункту (5).
4. Вычисление значений ошибок и исправление искаженных позиций по алгоритму Форни.
5. Восстановление информационных символов или выдача признака отказа от декодирования. Этап (5) при отсутствии отказа состоит в восстановлении информационных символов из полученного кодового слова. При систематическом коде РС этот этап сводится к выдаче символов на информационных позициях кода. Для несистематического кода выполняется преобразование, определяющее информационные элементы кода, например, дискретное преобразование Фурье.

8.2.1. Вычисление синдромов

Как и в случае двоичных кодов БЧХ, при декодировании кода РС синдромы вычисляются, как значения принятого полинома $r(x)$ в *нулях кода*.

$$S_j = r(\varepsilon^j), \tag{123}$$

где $j = b..b+2t-1$, $b = 0$ или $b = 1$, ε^j — корни порождающего полинома $g(x)$.

Предположим, что из канала связи получена комбинация рассмотренного ранее кода РС $(7,3)$ с образующим полиномом $g(x) = g_{(7,3),b=0}(x)$ (122), представляемая многочленом

$$r(x) = v(x) + e(x) = \varepsilon x^2 + \varepsilon^5 x^4, \tag{124}$$

тогда, согласно формуле (123) синдромы будут равны

$$\begin{aligned} S_0 &= r(1) = \varepsilon(1)^2 + \varepsilon^5(1)^4 = \varepsilon^6, \\ S_1 &= r(\varepsilon) = \varepsilon(\varepsilon)^2 + \varepsilon^5(\varepsilon)^4 = \varepsilon^5, \\ S_2 &= r(\varepsilon^2) = \varepsilon(\varepsilon^2)^2 + \varepsilon^5(\varepsilon^2)^4 = \varepsilon, \\ S_3 &= r(\varepsilon^3) = \varepsilon(\varepsilon^3)^2 + \varepsilon^5(\varepsilon^3)^4 = \varepsilon. \end{aligned} \tag{125}$$

8.2.2. Алгоритм Берлекэмпа-Мэсси

Открытый в 1968 году Э. Берлекэмпом и применённый к линейным кодам в 1969 году Дж. Мэсси [40, 41, 42], данный алгоритм синтезирует минимальный регистр с линейной обратной связью, порождающий заданную последовательность символов[36]. Рассмотрим вариант алгоритма Берлекэмпа-Мэсси, известный как алгоритм Мэсси [34].

Исходные данные: S_j, $j = b..b+2t-1$, $b = 0$ или $b = 1$, — синдромы.

Начальные условия: $\sigma(x) = 1$, корректор $\rho(x) = x$, счетчик $i = 0$, длина регистра $l = 0$.

1. Вычисляем различие $\Delta = \sum_{j=0}^{l} \sigma_j S_{i-j-1}$.
2. Проверяем различие Δ: если $\Delta = 0$, то переходим к (7).
3. Модифицируем многочлен локаторов ошибок: $\sigma_{new}(x) = \sigma(x) + \Delta\rho(x)$.
4. Проверяем длину регистра: если $2l \geq i$, то переходим к (6).
5. Исправляем длину регистра и заменяем корректор: $l = i - l$, $\rho(x) = \sigma(x)/\Delta$.
6. Обновляем многочлен локаторов ошибок: $\sigma(x) = \sigma_{new}(x)$.
7. Обновляем корректор: $\rho(x) = x\rho(x)$.
8. Обновляем счетчик: $i = i + 1$.

9. Если $i < d$, то переходим к (1), иначе останавливаемся.

Рассмотрим декодирование по алгоритму Мэсси на примере ранее рассмоpенного кода РС $(7,3)$ с образующим полиномом $g(x) = g_{(7,3),b=0}(x)$ (122) и полученной на вход декодера комбинации (124) [34]. Синдромы для выбранной комбинации получены в формуле (125).

$i = 1$:

$\sigma(x) = 1, \rho(x) = x, l = 0.$

$\Delta = \sum_{j=0}^{0} \sigma_j S_{1-j-1} = \sigma_0 S_0 = \varepsilon^6.$

$\sigma_{new}(x) = \sigma(x) + \Delta\rho(x) = 1 + \varepsilon^6 x, 2l = 0 < i.$

$l \to i - l = 1 - 0 = 1, \rho(x) = \sigma(x)/\Delta = 1/\varepsilon^6 = \varepsilon^{-6} = \varepsilon.$

$\sigma(x) = \sigma_{new}(x) = 1 + \varepsilon^6 x, \rho(x) \to x\rho(x) = \varepsilon x.$

$i = 2$:

$\Delta = \sum_{j=0}^{1} \sigma_j S_{2-j-1} = \sigma_0 S_1 + \sigma_1 S_0 = \varepsilon^5 + \varepsilon^6\varepsilon^6 = 0.$

$\rho(x) \to x\rho(x) = \varepsilon x^2.$

$i = 3$:

$\Delta = \sum_{j=0}^{1} \sigma_j S_{3-j-1} = \sigma_0 S_2 + \sigma_1 S_1 = \varepsilon + \varepsilon^6\varepsilon^5 = \varepsilon^2.$

$\sigma_{new}(x) = \sigma(x) + \Delta\rho(x) = 1 + \varepsilon^6 x + \varepsilon^3 x^2, 2l = 2 < i.$

$l \to i - l = 3 - 1 = 2, \rho(x) = \sigma(x)/\Delta = \varepsilon^5 + \varepsilon^4 x.$

$\sigma(x) = \sigma_{new}(x) = 1 + \varepsilon^6 x + \varepsilon^3 x^2, \rho(x) \to x\rho(x) = \varepsilon^5 x + \varepsilon^4 x^2.$

$i = 4$:

$\Delta = \sum_{j=0}^{2} \sigma_j S_{4-j-1} = \sigma_0 S_3 + \sigma_1 S_2 + \sigma_2 S_1 = \varepsilon + \varepsilon^6\varepsilon + \varepsilon^3\varepsilon^5 = 1.$

$\sigma_{new}(x) = \sigma(x) + \Delta\rho(x) = 1 + \varepsilon^6 x + \varepsilon^3 x^2 + \varepsilon^5 x + \varepsilon^4 x^2 = 1 + \varepsilon x + \varepsilon^6 x^2, 2l = 4 = i.$

$\sigma(x) = \sigma_{new}(x) = 1 + \varepsilon x + \varepsilon^6 x^2, \rho(x) \to x\rho(x) = \varepsilon^5 x^2 + \varepsilon^4 x^3.$

$i = 5 = d$: Стоп.

Теперь по ключевому уравнению найдем многочлен значений ошибок:

$\omega(x) = S(x)\sigma(x) \pmod{x^5} = 1 + \varepsilon^5 x + \varepsilon^3 x^2.$

8.2.3. Алгоритм Евклида

В основе этого алгоритма лежит процедура нахождения наибольшего общего делителя (НОД) двух полиномов. [34] Так, для решения ключевого уравнения, расширенный алгоритм Евклида применяется к многочленам

$r_0(x) = x^d$ и $r_{-1}(x) = S(x)$. Если на j-м шаге алгоритма получено решение

$$r_j(x) = a_j(x)x^d + b_j(x)S(x),$$

такое, что $\deg[r_j(x)] \leq (d-1)/2$, то $\omega(x) = r_j(x)$ и $\sigma(x) = b_j(x)$.

Расширенный алгоритм Евклида вычисления НОД состоит в следующем [34].

1. Вход: $r_0(x)$, $r_1(x)$, $\deg[r_0(x)] \geq \deg[r_1(x)]$.
2. Начальные условия: $a_0(x) = 1$, $b_0(x) = 0$, $a_1(x) = 0$, $b_1(x) = 1$.
3. На шаге j ($j \geq 2$), применяем длинное деление к многочленам $r_{j-2}(x)$ и $r_{j-1}(x)$, $r_{j-2}(x) = q_j(x)r_{j-1}(x) + r_j(x)$, $0 \leq \deg[r_j(x)] < \deg[r_{j-1}(x)]$.
4. Вычисляем $a_j(x) = a_{j-2}(x) - q_j(x)a_{j-1}(x)$, $b_j(x) = b_{j-2}(x) - q_j(x)b_{j-1}(x)$.
5. Останавливаем вычисления на итерации $last = j_{last}$, когда $\deg[r_{last}(x)] \leq r/2$.
6. Выход: НОД$(r_0x, r_1x) = r_k(x)$, где k наибольшее ненулевое целое такое, что $r_k(x) \neq 0$ и $k < j_{last}$.

Данный алгоритм отличается большей простотой аппаратной реализации по сравнению с алгоритмом Берлекэмпа-Мэсси, но требует больше операций в конечном поле [34].

Важно отметить, что алгоритм Евклида вычисляет $\sigma(x)$ и $\omega(x)$ одновременно, как $\sigma(x) = b_{last}(x)$ и $\omega(x) = r_{last}(x)$.

Рассмотрим декодирование по алгоритму Евклида на примере ранее рассмоpенного кода РС $(7,3)$ с образующим полиномом $g(x) = g_{(7,3),b=0}(x)$ (122) и полученной на вход декодера комбинации (124) [34]. Синдромы для выбранной комбинации получены в формуле (125).

Начинаем декодирование по алгоритму Евклида [34].

Начальные условия (шаг 1):

$$r_0(x) = x^5,\ r_1(x) = S(x) = 1 + \varepsilon^6 x + \varepsilon^5 x^2 + \varepsilon x^3 + \varepsilon x^4,\ b_0(x) = 0,\ b_1(x) = 1.$$

Шаг 2:

$$x^5 = (1 + \varepsilon^6 x + \varepsilon^5 x^2 + \varepsilon x^3 + \varepsilon x^4)(\varepsilon^6 x + \varepsilon^6) + \varepsilon^5 x^3 + x^2 + \varepsilon x + \varepsilon^6,$$
$$r_2(x) = \varepsilon^5 x^3 + x^2 + \varepsilon x + \varepsilon^6,$$
$$q_2(x) = \varepsilon^6 x + \varepsilon^6,$$
$$b_2(x) = 0 + (\varepsilon^6 x + \varepsilon^6)(1) = \varepsilon^6 x + \varepsilon^6.$$

Шаг 3:

$$1 + \varepsilon^6 x + \varepsilon^5 x^2 + \varepsilon x^3 + \varepsilon x^4 = (\varepsilon^5 x^3 + x^2 + \varepsilon x + \varepsilon^6)(\varepsilon^3 x + \varepsilon^2) + \varepsilon^6 x^2 + \varepsilon x + \varepsilon^3,$$
$$r_3(x) = \varepsilon^6 x^2 + \varepsilon x + \varepsilon^3,$$
$$q_3(x) = \varepsilon^3 x + \varepsilon^2,$$
$$b_3(x) = 1 + (\varepsilon^3 x + \varepsilon^2)(\varepsilon^6 x + \varepsilon^6) = \varepsilon^3 + \varepsilon^4 x + \varepsilon^2 x^2.$$

Алгоритм останавливается, так как $\deg[r_3(x)] = 2 = r/2$.

Следовательно:

$\sigma(x) = \varepsilon^3 + \varepsilon^4 x + \varepsilon^2 x^2 = \varepsilon^3(1 + \varepsilon x + \varepsilon^6 x^2)$,

$\omega(x) = \varepsilon^3 + \varepsilon x + \varepsilon^6 x^2 = \varepsilon^3(1 + \varepsilon^5 x + \varepsilon^3 x^2)$.

8.2.4. Процедура Ченя

Как и в случае двоичных кодов БЧХ, для поиска корней $\sigma(x)$ на множестве локаторов позиций кодовых символов используется метод метод Ченя, т. е. для всех ненулевых элементов $\beta \in \mathrm{GF}(2^m)$, проверяется условие $\sigma(\beta^{-1}) = 0$. [34]

Для примера, рассмотренного выше, решением ключевого уравнения будут элементы поля ε^{-2} и ε^{-4}, т. е. $\beta_{j_1} = \varepsilon^2$ и $\beta_{j_2} = \varepsilon^4$.

$$\sigma(x) = 1 + \varepsilon x + \varepsilon^6 x^2 = (1 + \varepsilon^2 x)(1 + \varepsilon^4 x).$$

Таким образом, ошибки произошли на позициях $j_1 = 2$ и $j_2 = 4$.

8.2.5. Алгоритм Форни

Согласно алгоритму Форни значения ошибок вычисляются по формуле

$$e_{j_l} = \beta_{j_l}^{1-b} \omega(\beta_{j_l}^{-1})[\sigma'(\beta_{j_l}^{-1})]^{-1}. \tag{126}$$

Для рассматриваемого примера это уравнение будет иметь вид

$$e_{j_l} = \beta_{j_l}(1 + \varepsilon^5 \beta_{j_l}^{-1} + \varepsilon^3 \beta_{j_l}^{-2})[\varepsilon]^{-1}$$

тогда

$$e_2 = \varepsilon^2(1 + \varepsilon^5 \varepsilon^{-2} + \varepsilon^3 \varepsilon^{-4})\varepsilon^{-1} = \varepsilon,$$
$$e_4 = \varepsilon^4(1 + \varepsilon^5 \varepsilon^{-4} + \varepsilon^3 \varepsilon^{-8})\varepsilon^{-1} = \varepsilon^5.$$

Таким образом, $e(x) = \varepsilon x^2 + \varepsilon^5 x^4$ и декодированное слово равно

$$\hat{v}(x) = r(x) + e(x) = 0$$

Исправлены две ошибки.

8.3. Эквивалентные коды РС

Эквивалентные (n,k) коды Рида-Соломона строятся над полем $\mathrm{GF}(2^k)$ с некоторым порождающим поле многочленом $p(x)$ k-ой степени, корнем которого будет в общем случае первообразный элемент $\varepsilon \in \mathrm{GF}(2^k)$ [17]. В отличие от кодов РС в узком смысле, для построения эквивалентного циклического

кода Рида-Соломона, в дальнейшем кода РСЭ, выбирается образующий многочлен

$$g(x) = \prod_{i=1}^{k}(x + \varepsilon_i), \tag{127}$$

где ε_i — корни образующего многочлена.

Код РСЭ (n,k) является эквивалентным для (n,k) кода РС в узком смысле. Понятие эквивалентных кодов введено в [17]. Эквивалентность заключается в том, что оба кода имеют одинаковые параметры n и k, а также одинаковую исправляющую способность, что показано далее. Код РСЭ также является дуальным. Дуальность кодов заключается в том, что код РСЭ (n,k), как и (n,k) код РС, дуален по параметрам коду РС $(n,n-k)$. Нужно отметить, что образующий полином кода РС в узком смысле $g_{\text{РС}}$ имеет степень $n-k$, а образующий полином кода РСЭ $g_{\text{РСЭ}}$ имеет степень k, и совпадает с образующим полиномом дуального кода РС $(n,n-k)$. При этом справедливо следующее выражение [1]:

$$g_{\text{РСЭ}} = \frac{x^n - 1}{g_{\text{РС}}}, \qquad n = 2^k - 1.$$

Существует два варианта кодов РСЭ. В первом используются так называемые несопряжённые корни, когда $\varepsilon_i = \varepsilon^{i-1}$ или $\varepsilon_i = \varepsilon^i$. Во втором случае, корни образующего полинома должны соответствовать одному или нескольким циклотомическим классам поля, над которым построен код. Такие корни являются сопряженными. При этом образующий полином РСЭ-кода $g(x)$ равен одному из минимальных многочленов над полем либо равен произведению нескольких таких многочленов.

Многочлен $g(x)$ является характеристическим многочленом

$$g(x) = g_0x^k + g_1x^{k-1} + g_2x^{k-2} + \ldots + g_{k-1}x + g_k; \quad g_i \in \text{GF}(2^k), \tag{128}$$

порождающим возвратные рекурсивные последовательности $\{s\} = (s_0 \;\; s_1 \;\; s_2 \;\; s_3 \;\; \ldots \;\; s_{2^k-2})$ с периодом $n = 2^k - 1$ и элементами, принадлежащими полю $\text{GF}(2^k)$ [17].

В соответствии с видом характеристического многочлена $g(x)$, у которого всегда $g_0 = 1$, любая возвратная последовательность $\{s\}$ будет удовлетворять рекуррентному уравнению:

$$s_i = g_1s_{i-1} + g_2s_{i-2} + \ldots + g_{k-1}s_{i-k+1} + g_ks_{i-k}, \tag{129}$$

где $i \geq k$ и, кроме того, индекс i приводится по модулю $2^k - 1$.

Как и в случае кодов РС, кодирующее устройство кода РСЭ может строиться по принципу систематического или несистематического кодирования.

В случае систематического кодирования кода РСЭ информационными элементами кода (n,k) будут начальные k элементов последовательности $\{s\}$, т. е. $s_0 s_1 \ldots s_{k-1}$ [17]. Остальные избыточные элементы кодовой комбинации, как рекуррентной последовательности, определяются по рекуррентной формуле (129). Блок-схема реализации кодирующего устройство такого систематического кода представлена на рисунке 8.3.

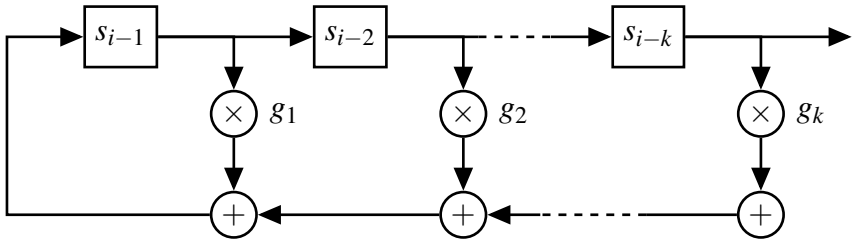

Рис. 8.3. Блок-схема кодирующего устройства систематического кода РСЭ

В основе несистематического кодирования кодов РСЭ лежит тот факт, что кодовые комбинации кода РСЭ являются возвратными последовательностями [17]. Как известно [17], произвольный элемент возвратной последовательности может быть выражен через корни $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$ характеристического многочлена $g(x)$ и соответствующие корням элементы поля $A_1, A_2, \ldots, A_k$ как

$$s_i = A_1\varepsilon_1^i + A_2\varepsilon_2^i + \ldots + A_k\varepsilon_k^i. \tag{130}$$

В случае несистематического кодирования элементы поля $A_1, A_2, \ldots, A_k$ играют роль информационных элементов и процесс кодирования осуществляется в соответствии с формулой (130). Блок-схема кодирующего устройства кода РСЭ для случая несистематического кодирования представлена на рисунке 8.4. Как можно видеть, реализация кодирующего устройства несистематического кода имеет простую реализацию. Информационные элементы A_1, $A_2, \ldots, A_k$, записанные как исходные в k-разрядные ячейки памяти с обратной связью, порождают на выходе кодовую последовательность $\{s\}$.

Далее определим исправляющую способность кодов РСЭ. Поскольку ранее было показано, что существует два вида кодов РСЭ: с несопряженными и с сопряженными корнями образующего многочлена $g(x)$, расмотрим их отдельно и сравним их исправляющие способности.

Для определения исправляющей способности кода необходимо определить его весовой спектр, т. е. распределение весов для кода. Для определения количества кодовых слов с весом ω можно использовать два способа. Первый способ — расчет распределения весов по формуле (131) [41, 19].

$$A(\omega) = c_n^{\omega}(q-1)\sum_{j=0}^{\omega-d_{\min}}(-1)^j c_{\omega-1}^{j} q^{\omega-j-d_{\min}}. \tag{131}$$

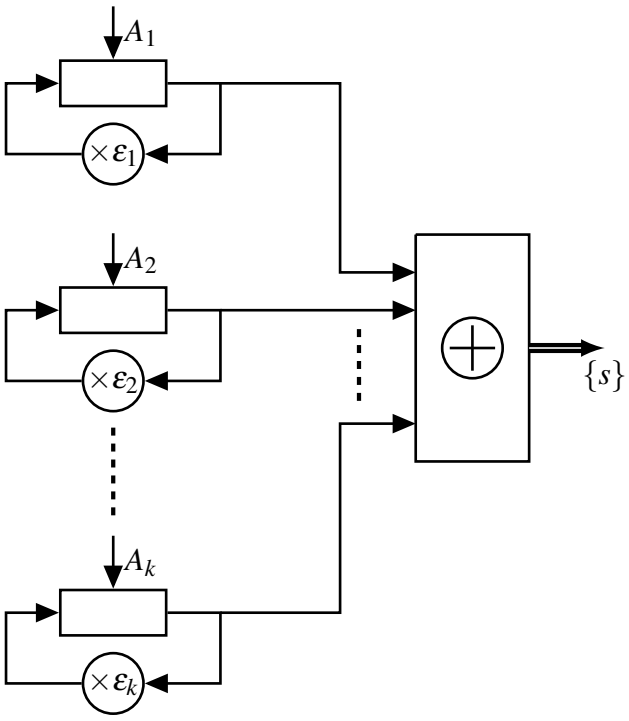

Рис. 8.4. Блок-схема кодирующего устройства несистематического кода РСЭ

где ω — вес кодового слова, а $A(\omega)$ — количество кодовых слов с весом ω.

Второй способ — экспериментальный. Он заключается в переборе всех возможных кодовых слов и подсчете их весов. Распределение весов, полученное вышеприведенными способами совпадает.

Приведем диаграммы распределения весов для кодов РСЭ $(15,4)$ и $(31,5)$. Для примера возьмем следующие коды:

1. Код РСЭ $(15,4)$ с сопряженными корнями над полем $\mathrm{GF}(2^4)$:
   $p(x) = x^4 + x + 1$
   $g(x) = (x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^4)(x+\varepsilon^8) = x^4 + x + 1.$
2. Код РСЭ $(15,4)$ с несопряженными корнями над полем $\mathrm{GF}(2^4)$:
   $p(x) = x^4 + x + 1;$
   $g(x) = (x+1)(x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^3) = x^4 + \varepsilon^{12}x^3 + \varepsilon^4 x^2 + x + \varepsilon^6.$
3. Код РСЭ $(31,5)$ с сопряженными корнями над полем $\mathrm{GF}(2^5)$:
   $p(x) = x^5 + x^3 + 1;$
   $g(x) = (x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^4)(x+\varepsilon^8)(x+\varepsilon^{16}) = x^5 + x^3 + 1.$
4. Код РСЭ $(31,5)$ с несопряженными корнями над полем $\mathrm{GF}(2^5)$:
   $p(x) = x^5 + x^2 + 1;$
   $g(x) = (x+1)(x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^3)(x+\varepsilon^4) =$
   $= x^5 + \varepsilon^{15}x^4 + \varepsilon^{21}x^3 + \varepsilon^{23}x^2 + \varepsilon^{21}x + \varepsilon.$

Как можно видеть из рисунка 8.5, код РСЭ $(15,4)$ с сопряженными корнями имеет минимальное кодовое расстояние $d_{\min} = 8$ и, соответственно, кратность исправляемых ошибок $t = 3$. Для кода РСЭ $(15,4)$ с несопря-

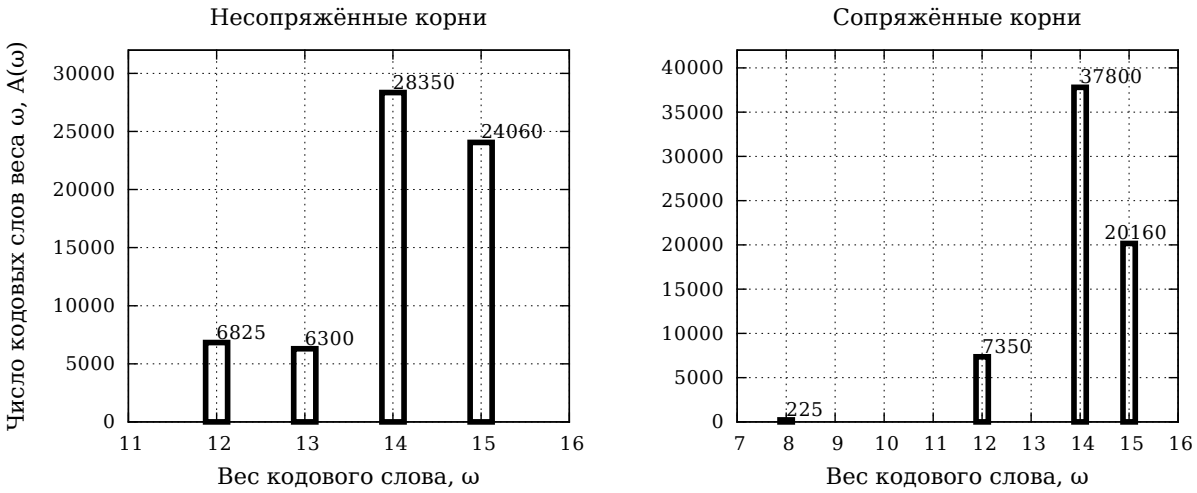

Рис. 8.5. Распределение весов для кода РСЭ $(15,4)$

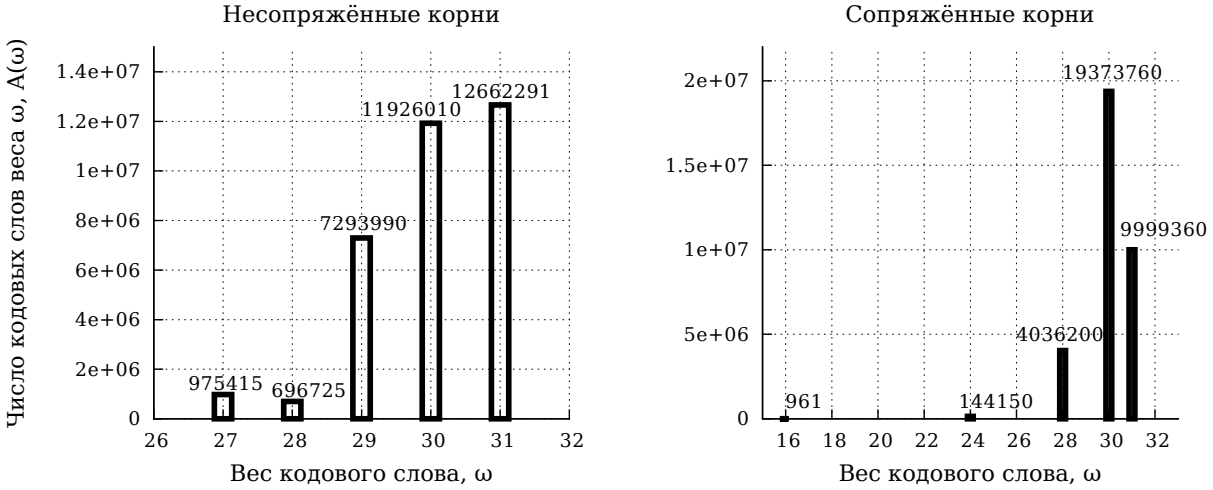

Рис. 8.6. Распределение весов для кода РСЭ $(31,5)$

женными корнями минимальное кодовое расстояние $d_{\min} = 12$, а кратность исправляемых ошибок $t = 5$, что, как следует из формулы 116 совпадает с теоретически-достижимой кратностью для кодов РС $(15,4)$.

Аналогичная ситуация прослеживается и для кодов РСЭ $(31,5)$. Как видно из рисунка 8.6, код РСЭ $(31,5)$ с сопряженными корнями имеет минимальное кодовое расстояние $d_{\min} = 16$ и кратность исправляемых ошибок $t = 7$. Для кода РСЭ $(31,5)$ с несопряженными корнями минимальное кодовое расстояние $d_{\min} = 27$, а кратность исправляемых ошибок $t = 13$, что совпадает с теоретически-достижимой кратностью для кодов РС $(31,5)$.

Таким образом, минимальное кодовое расстояние $d_{\min}$ РСЭ кодов с сопряжёнными корнями меньше, нежели минимальное кодовое расстояние кодов с несопряжёнными корнями.

8.4. Определение информационных элементов по k-элементному участку кодовой последовательности кода РСЭ

В разделе 8.3 было показано, что кодовые комбинации (n,k) РСЭ-кода представляют из себя возвратные рекурсивные последовательности $\{s\}$ с периодом $n = 2^k - 1$ и элементами, принадлежащими полю GF(2^k). Известно [17], что произвольный элемент возвратной последовательности может быть выражен через корни $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_k$ характеристического многочлена $g(x)$ и соответствующие корням элементы поля $A_1, A_2, \ldots, A_k$ по формуле (130).

В [17] показано, что элементы $A_1, A_2, \ldots, A_k$ могут быть определены, по любому k-элементному участку $\{s_i, s_{i+1}, \ldots, s_{i+k-1}\}$ замкнутой в кольцо кодовой последовательности $\{s\}$ в соответствии с выражениями:

$$\begin{aligned} A_1 &= \varepsilon_1^{-i}(\alpha_{11}s_i + \alpha_{12}s_{i+1} + \ldots + \alpha_{1k}s_{i+k-1}); \\ A_2 &= \varepsilon_2^{-i}(\alpha_{21}s_i + \alpha_{22}s_{i+1} + \ldots + \alpha_{2k}s_{i+k-1}); \\ &\ldots \\ A_k &= \varepsilon_k^{-i}(\alpha_{k1}s_i + \alpha_{k2}s_{i+1} + \ldots + \alpha_{kk}s_{i+k-1}); \end{aligned} \tag{132}$$

где постоянные коэффициенты α_{ij} $i, j = 1, 2, \ldots, k$, зависят от характеристического многочлена $g(x)$ и определяются по выражению

$$\alpha_{ij} = \frac{\sum_{l=0}^{k-j} g_{k-j-l}\varepsilon_i^l}{g'(\varepsilon_i)}; \quad \text{GF}(2^k) \tag{133}$$

через корни и коэффициенты g_j многочлена $g(x)$.

При этом необходимо отметить, что в случае использования РСЭ кода, корни образующего полинома которого являются сопряженными элементами поля ε, ε^2, $\ldots$, $\varepsilon^{2^{k-1}}$, образующий полином РСЭ-кода $g(x)$ совпадает с образующим полиномом поля GF(2^k) $p(x)$, а коэффициенты α_{1j} совпадают с элементами двойственного базиса поля Галуа α_j, рассчитываемыми по формуле (7). Коэффициенты α_{ij} можно определить по формуле:

$$\alpha_{ij} = \alpha_j^{2^{i-1}}.$$

Таким образом, для кодов РСЭ с сопряженными корнями, элементы A_1, $A_2, \ldots, A_k$ могут быть вычислены по формуле:

$$A_j = \varepsilon_j^{-i}(\alpha_1^{2^{j-1}}s_i + \alpha_2^{2^{j-1}}s_{i+1} + \ldots + \alpha_k^{2^{j-1}}s_{i+k-1}), \tag{134}$$

где $\alpha_1, \ldots, \alpha_k$ — элементы двойственного базиса поля Галуа, на основе которого образован код РСЭ.

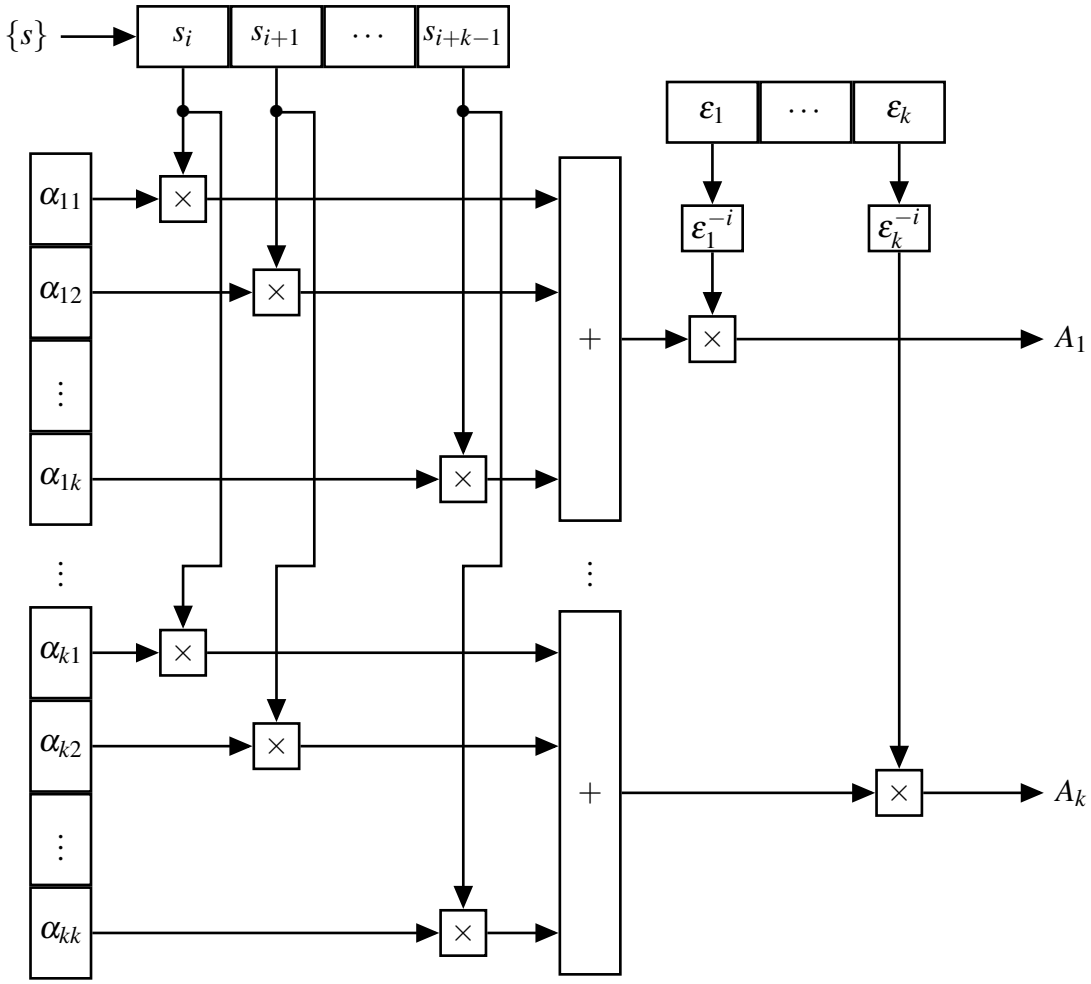

Рис. 8.7. Функциональная схема устройства вычисления информационных элементов согласно формулам (132)

На рисунке 8.7 представлена функциональная схема устройства вычисления информационных элементов согласно формулам (132).

Таким образом, получается, что для определения информационных элементов кодовой последовательности $\{s\}$ (n,k) кода РСЭ достаточно знать любые k последовательных элементов данной последовательности, замкнутой в кольцо, из которых по формуле (132) или (134) можно определить элементы $A_1, A_2, \ldots, A_k$. Эти элементы для несистематических кодов РСЭ сами могут являться информационными элементами, а в случае систематических кодов РСЭ также однозначно определяют информационные элементы в составе рекуррентной последовательности $\{s\}$.

Описанные выше свойства кодовых комбинаций кода РСЭ позволяют организовать декодирование по мажоритарному принципу. Далее рассмотрим процесс декодирования.

Для простоты будем считать, что используется несистематический код РСЭ (n,k), для которого рассчитываемые по формулам (132) или (134) элементы $A_1, A_2, \ldots, A_k$ являются информационными.

Вначале рассмотрим процесс декодирования кодовой комбинации $\{s\}$, которая не содержит ошибок.

Очевидно, что перебрав все n k-элементных участков замкнутой в кольцо безошибочной кодовой комбинации $\{s\}$ и проведя над ними вычисления по формуле (132), мы получим n одинаковых наборов информационных элементов $A_1, A_2, \ldots, A_k$.

Теперь рассмотрим кодовую комбинацию, содержащую одну ошибку.

В этом случае получится, что k k-элементных участков замкнутой в кольцо кодовой комбинации $\{s\}$ при проведени расчётов по формуле (132) дадут результат, отличающийся от реального набора информационных элементов. Мы получим $n-k$ одинаковых наборов информационных элементов, соответствующих исходной информационной комбинации, и k отличающихся наборов. В итоге можно построить диаграмму накопления результатов, которая в работах [17, 43] названа «лес». В дальнейшем будем употреблять именно это название. В общем виде «лес» представлен на рисунке 8.8.

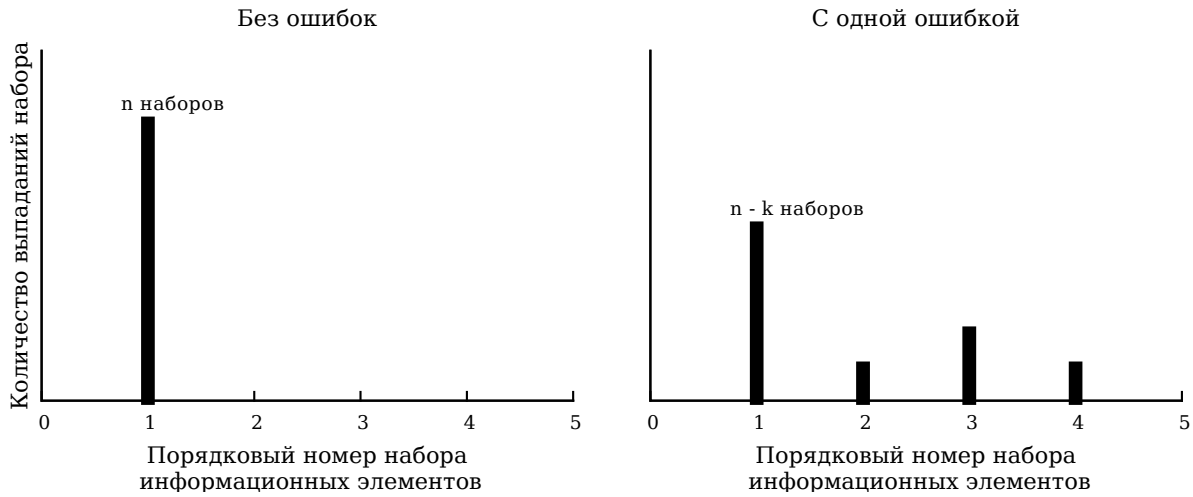

Рис. 8.8. Общий вид диаграммы «лес»

Согласно принятым в работах [17, 43] обозначениям, будем называть «деревьями» столбцы диаграммы вида «лес».

Далее рассмотрим изменение диаграммы «лес» при росте количества ошибок на примере реальной кодовой комбинации.

Для примера возьмём РСЭ код $(15,4)$ с несопряжёнными корнями, построенный над полем Галуа $\mathrm{GF}(2^4)$, образованным полиномом $p(x) = x^4 + x + 1$.

Вначале определим образующий полином кода по формуле (127).

$$g(x) = \prod_{i=1}^{4}(x+\varepsilon^i) = x^4 + \varepsilon^{12}x^3 + \varepsilon^4 x^2 + x + \varepsilon^6.$$

Пусть информационная последовательность $\{f\}$ равна:

$$\{f\} = \left(\varepsilon^2\varepsilon^6\varepsilon^8\varepsilon^{10}\right)$$

тогда закодированная несистематическим кодером (см. рис. 8.4) кодовая последовательность $\{s\}$ будет равна:

$$\{s\} = \left(\varepsilon^9\varepsilon^8\varepsilon^6\varepsilon^2\varepsilon^9\varepsilon^8 00\varepsilon^4 0\varepsilon^6\varepsilon^6\varepsilon^9\varepsilon^{10}\varepsilon^8\right).$$

Наложим на указанную кодовую комбинацию поочерёдно однократную, двукратную и трёхкратную ошибки и декодируем полученные последовательности по методу МДБ. Полученный в результате «лес» представлен на рисунке 8.9.

Как можно видеть из рисунка 8.9, при увеличении кратности ошибки доля выпадений «правильного» набора сокращается и соответствующее «дерево» уменьшается по высоте. Одновременно с этим растёт число выпадений «неправильных» наборов, что отображено на диаграмме увеличением количества боковых «деревьев».

Данный процесс приводит к тому, что при увеличении кратности накладываемой ошибки на диаграмме высота «дерева», соответствующего «правильному» набору, может уменьшиться настолько, что не представляется возможным выделить его из других деревьев по максимальному значению.

Анализируя приведённые диаграммы и материалы представленные в работах [17, 43], можно отметить, что, в случае отсутствия ошибок или малой их кратности, «дерево», соответствующее «правильному набору», имеет высоту значительно превосходящую высоту боковых деревьев (в случае их наличия). Учитывая, что суммарная высота всех «деревьев» не может быть больше n, очевидно, что для выделения «дерева», однозначно соответствующего «правильному» набору, его высота должна составлять не менее $\lfloor \frac{n}{2} \rfloor + 1$.

Поскольку диаграмма «лес» строится последовательно по мере обработки k-элементных участков, можно сказать, что как только на диаграмме появляется дерево высотой равное $\lfloor \frac{n}{2} \rfloor + 1$, процесс декодирования комбинации можно прекращать. Следовательно, в случае декодирования комбинации, не содержащей ошибки, достаточно обработать лишь $\lfloor \frac{n}{2} \rfloor + 1$ k-элементных участков, а значит, учитывая что элементы кодовой комбинации принимаются приёмником последовательно, процесс декодирования может быть закончен ещё до приёма всей кодовой комбинации.

Для рассматриваемого на рисунке 8.9 примера данная величина будет равна:

$$\left\lfloor \frac{n}{2} \right\rfloor + 1 = \left\lfloor \frac{15}{2} \right\rfloor + 1 = 8.$$

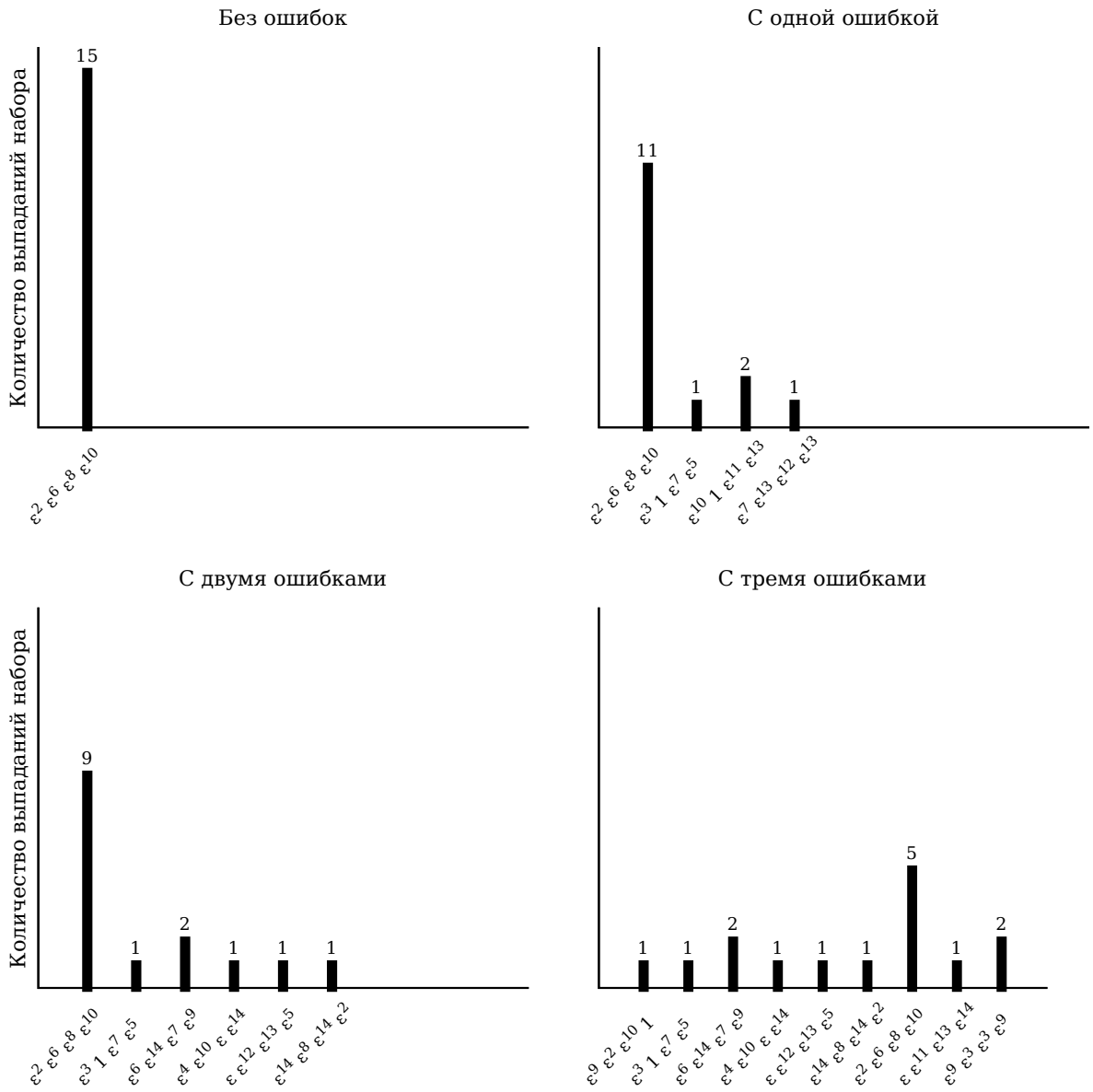

Рис. 8.9. Пример изменения вида диаграммы «лес» при росте кратности ошибки

8.5. Декодирование с использованием децимаций

В работах [17, 43] показано, что корректирующие свойства данного алгоритма могут быть расширены путём использования так называемых децимаций с индексами $q = p^j$, где $j = 1, \ldots, (k-1)$.

Децимации представляют из себя перемежение элементов кодовой последовательности по определённому правилу. При этом, возможность применения децимаций к кодовой последовательности предъявляет определённые требования к образующему многочлену кода РСЭ $p(x)$. Эти требования возможно сформулировать в виде следующего свойства:

Комбинации циклического кода РСЭ могут быть мажоритарно декодированы с использованием децимаций, если их порождающий много-

член $p(x)$ представляет собой один или произведение нескольких многочленов $f_i(x)$, входящих в разложение двучлена $(x^{p^k-1}-1)$ на неприводимые многочлены деления круга [17].

Этому условию удовлетворяют рассмотренные в разделе 8.3 коды РСЭ с сопряжёнными корнями полинома.

В случае выполнения этого требования будут справедливы две теоремы, названные в [17] теорема идентичности и теорема однозначности.

Теорема идентичности. Если характеристический многочлен $p(x)$ представляет собой один или произведение нескольких минимальных многочленов, входящих в разложение двучлена $(x^{p^m-1}-1)$, то исходная рекуррентная последовательность $\{s\} = (s_0 s_1 s_2 \ldots s_{p^m-2})$, образованная по многочлену $p(x)$, и последовательности $\{u\} = (u_0 u_1 u_2 \ldots u_{p^m-2})$, полученные из $\{s\}$ путём децимаций с индексом $q = p^j$, где $j = 1, 2, \ldots, (m-1)$, будут удовлетворять одному и тому же рекуррентному уравнению (соотношению).

Теорема однозначности. Если элементы $A_1, A_2, A_3, \ldots \in \mathrm{GF}(p^m)$, соответствующие сомножителям характеристического многочлена $p(x)$, однозначно определяют начальную фазу рекуррентной последовательности $\{s\}$, то начальная фаза последовательностей $\{u\}$, полученных из последовательности $\{s\}$ путём её децимации с индексом $q = p^j$, где $j = 1, 2, \ldots, (m-1)$, также будет однозначно определяться теми же элементами $A_1, A_2, A_3, \ldots$, но имеющими циклический сдвиг вправо на j шагов.

Подробные доказательства данных теорем приведены в [17].

Рассмотрим пример использования процедуры децимирования при декодировании кодовой комбинации кода РСЭ.

Возьмём код РСЭ $(7,3)$ с сопряжёнными корнями, построенный над полем Галуа $\mathrm{GF}(2^3)$, образованным полиномом $p(x) = x^3 + x + 1$.

Согласно формуле (127), образующий полином кода будет равен:

$$g(x) = (x+\varepsilon)(x+\varepsilon^2)(x+\varepsilon^4) = x^3 + x + 1.$$

Пусть информационная последовательность $\{f\}$ равна:

$$\{f\} = \left(\varepsilon^6 \varepsilon \varepsilon^4\right)$$

тогда закодированная несистематическим кодером (см. рис. 8.4) кодовая последовательность $\{s\}$ будет равна:

$$\{s\} = \left(10\varepsilon 1 \varepsilon \varepsilon^3 \varepsilon^3\right).$$

Учитывая то, что используется код РСЭ с сопряжёнными корнями, воспользуемся формулой (6) для нахождения элементов двойственного базиса

[ht]

Таблица 8.1

Децимирование кодовой последовательности $\{s\}$ *для кода* $(7,3)$

Индекс децимации	Децимированная комбинация
$q=2^0=1$	$\{u\}_1=\{s\}=\begin{Bmatrix}\{s_0 & s_1 & s_2 & s_3 & s_4 & s_5 & s_6\}\\ (1 & 0 & \varepsilon & 1 & \varepsilon & \varepsilon^3 & \varepsilon^3)\end{Bmatrix}$
$q=2^1=2$	$\{u\}_2=\begin{Bmatrix}\{u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6\}\\ \{s_0 & s_2 & s_4 & s_6 & s_1 & s_3 & s_5\}\\ (1 & \varepsilon & \varepsilon & \varepsilon^3 & 0 & 1 & \varepsilon^3)\end{Bmatrix}$
$q=2^2=4$	$\{u\}_4=\begin{Bmatrix}\{u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6\}\\ \{s_0 & s_4 & s_1 & s_5 & s_2 & s_6 & s_3\}\\ (1 & \varepsilon & 0 & \varepsilon^3 & \varepsilon & \varepsilon^3 & 1)\end{Bmatrix}$

поля Галуа α_j:

$$\{\alpha_j\}=\left(1\varepsilon^2\varepsilon\right).$$

Проведя расчёты по формулам (134), можно убедиться, что при переборе всех k-элементных участков последовательности $\{s\}$, в результате получается исходная информационная последовательность $\{f\}$.

Далее проведём децимации рассматриваемой кодовой комбинации. Результат можно представить в виде таблицы 8.1.

Теперь покажем, что перебрав все k-элементные участки децимированных комбинаций $\{u\}$, и подставив их в формулу (134), мы получим те же коэффициенты $A_1, A_2, \ldots, A_k$, соответствующие исходной информационной последовательности $\{f\}$, лишь сдвинутые циклически на 1 и 2 шага соответственно.

Возьмём для примера участок децимированной комбинации $\{u\}_2$

$$\{u_1u_2u_3\}=\{s_2s_4s_6\}=(\varepsilon\varepsilon\varepsilon^3).$$

Тогда, в соответствии с формулой (134) и теоремой однозначности получим:

$$\begin{aligned}
A_{21}&=\varepsilon_1^{-1}(\varepsilon\alpha_1+\varepsilon\alpha_2+\varepsilon^3\alpha_3)=\varepsilon^{-1}(\varepsilon+\varepsilon^3+\varepsilon^4)=\varepsilon^4=A_{13}\\
A_{22}&=\varepsilon_2^{-1}(\varepsilon\alpha_1^2+\varepsilon\alpha_2^2+\varepsilon^3\alpha_3^2)=\varepsilon^{-2}(\varepsilon+\varepsilon^5+\varepsilon^5)=\varepsilon^6=A_{11}\\
A_{23}&=\varepsilon_3^{-1}(\varepsilon\alpha_1^4+\varepsilon\alpha_2^4+\varepsilon^3\alpha_3^4)=\varepsilon^{-4}(\varepsilon+\varepsilon^2+1)=\varepsilon=A_{12}
\end{aligned}$$

Аналогично рассмотрим k-элементный участок децимированной комбинации $\{u\}_4$

$$\{u_1u_2u_3\}=\{s_4s_1s_5\}=(\varepsilon0\varepsilon^3).$$

[ht]

Таблица 8.2

Результаты декодирования кодовой комбинации кода $(7,3)$ *с двумя ошибками*

Значения коэффициентов			Количество сочетаний A_1, A_2, A_3 при обработке последовательности с индексом децимации q			Суммарное количество сочетаний A_1, A_2, A_3
A_1	A_2	A_3	$q=1(j=0)$	$q=2(j=1)$	$q=4(j=2)$	
ε^6	ε	ε^4	1	2	3	6
1	ε^4	ε	1	1	1	3
ε^3	ε^2	0	2	0	0	2
ε	ε^2	1	1	0	0	1
ε	0	ε^5	2	0	0	2
ε^2	1	ε	0	2	0	2
ε^6	ε^2	ε^2	0	1	0	1
ε^5	ε^3	1	0	1	0	1
0	0	1	0	0	1	1
1	ε^5	ε^3	0	0	1	1
ε^4	ε	1	0	0	1	1

В соответствии с формулой (134) и теоремой однозначности получим:

$$A_{41} = \varepsilon_1^{-1}(\varepsilon\alpha_1 + \varepsilon^3\alpha_3) = \varepsilon^{-1}(\varepsilon + \varepsilon^4) = \varepsilon = A_{12}$$
$$A_{42} = \varepsilon_2^{-1}(\varepsilon\alpha_1^2 + \varepsilon^3\alpha_3^2) = \varepsilon^{-2}(\varepsilon + \varepsilon^5) = \varepsilon^4 = A_{13}$$
$$A_{43} = \varepsilon_3^{-1}(\varepsilon\alpha_1^4 + \varepsilon^3\alpha_3^4) = \varepsilon^{-4}(\varepsilon + 1) = \varepsilon^6 = A_{11}$$

Аналогичные результаты получаются при проведении вычислений по всем k-элементным участкам децимированных комбинаций $\{u\}_2$ и $\{u\}_4$.

Теперь введём в кодовую комбинацию ошибки и покажем, что использование децимаций позволяет повысить эффективность декодирования.

Пусть была принята комбинация $\{h\}$, содержащая две ошибки:

$$\{h\} = \left(\varepsilon^5 0 \varepsilon\varepsilon 5 \varepsilon\varepsilon^3\varepsilon^3\right).$$

Обработаем по всем k-элементным участкам ($k=3$) последовательность $\{h\}$ и две её децимированные последовательности так, как показано выше. Полученные коэффициенты A_1, A_2 и A_3 сведём в таблицу 8.2.

Для большей наглядности представим результаты вычислений в виде набора диаграмм «лес» на рисунке 8.10.

Как можно видеть из таблицы 8.2 и рисунка 8.10, обработка только одной последовательности $\{h\}$ без децимаций не позволяет по высоте «деревьев» выделить правильный набор коэффициентов. Однако, при обработке всех трёх последовательностей, сводные результаты позволяют однозначно определить информационные элементы.

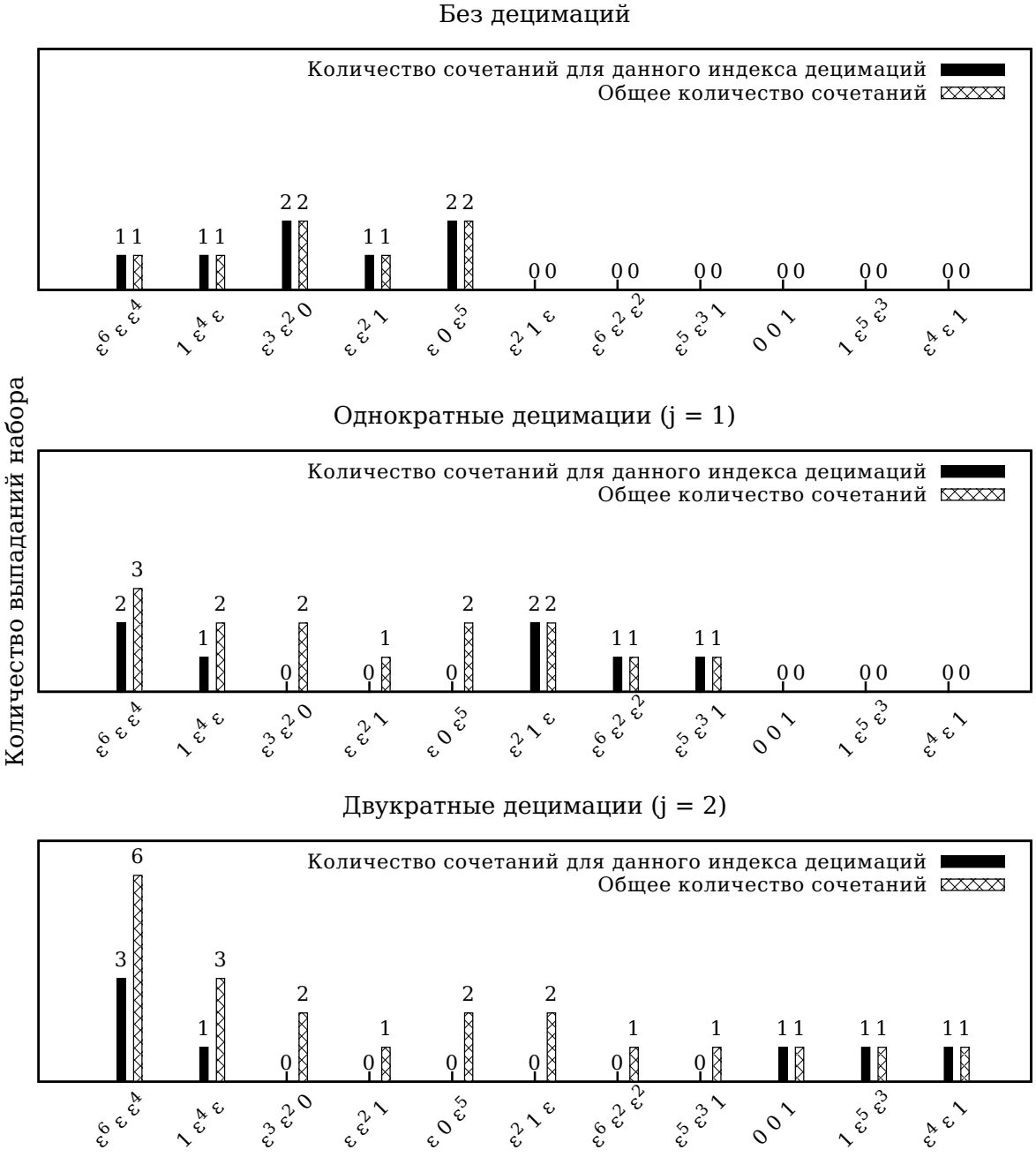

Рис. 8.10. Результаты декодирования кодовой комбинации кода $(7,3)$ *с двумя ошибками в виде диаграммы «лес»*

Таким образом, этот пример позволяет сделать предварительный вывод о том, что применение децимаций позволяет усилить корректирующие свойства кодов РСЭ, удовлетворяющих условиям теорем идентичности и однозначности.

Далее рассмотрим пример применения децимаций для кода $(31,11)$ с сопряжёнными корнями, построенного над полем Галуа $\mathrm{GF}(2^5)$, образованным полиномом $p(x) = x^5 + x^2 + 1$.

[ht]

Таблица 8.3

Результаты декодирования децимированных кодовых комбинаций кода $(31,11)$

Индекс децимации	Декодированная комбинация										
1	1	ε	ε^{18}	ε^{2}	ε^{5}	ε^{19}	ε^{11}	ε^{3}	ε^{29}	ε^{6}	ε^{27}
2	ε^{5}	1	ε	ε^{18}	ε^{2}	ε^{6}	ε^{19}	ε^{11}	ε^{3}	ε^{29}	ε^{27}
4	ε^{2}	ε^{5}	1	ε	ε^{18}	ε^{29}	ε^{6}	ε^{19}	ε^{11}	ε^{3}	ε^{27}
8	ε^{18}	ε^{2}	ε^{5}	1	ε	ε^{3}	ε^{29}	ε^{6}	ε^{19}	ε^{11}	ε^{27}
16	ε	ε^{18}	ε^{2}	ε^{5}	1	ε^{11}	ε^{3}	ε^{29}	ε^{6}	ε^{19}	ε^{27}

Пусть образующий полином кода, согласно теореме идентичности, равен произведению минимальных многочленов $f_1(x) = x^5 + x^2 + 1$, $f_2(x) = x^5 + x^4 + x^3 + x^2 + 1$ и $f_3(x) = x + 1$, входящих в разложение двучлена $x^{31} - 1$:

$$\begin{aligned} g(x) &= (x^5 + x^2 + 1)(x^5 + x^4 + x^3 + x^2 + 1)(x + 1) = \\ &= x^{11} + x^8 + x^7 + x^5 + x^4 + x^3 + x + 1. \end{aligned}$$

В этом случае, корнями полинома являются элементы выбранного поля Галуа соответствующие выбранным минимальным многочленам:

$$\left[\varepsilon\,\varepsilon^2\,\varepsilon^4\,\varepsilon^8\,\varepsilon^{16}, \varepsilon^6\,\varepsilon^{12}\,\varepsilon^{24}\,\varepsilon^{17}\,\varepsilon^3, \varepsilon^0\right].$$

Пусть информационная последовательность $\{f\}$ равна:

$$\{f\} = \left(1\varepsilon\varepsilon^{18}\varepsilon^2\varepsilon^5\varepsilon^{19}\varepsilon^{11}\varepsilon^3\varepsilon^{29}\varepsilon^6\varepsilon^{27}\right)$$

тогда закодированная несистематическим кодером (см. рис. 8.4) кодовая последовательность $\{s\}$ будет равна:

$$\begin{aligned} \{s\} = \; & \left(0\varepsilon^5\varepsilon^5\varepsilon^{28}\varepsilon^7\varepsilon^{24}\varepsilon^{30}\varepsilon^{30}\varepsilon^{21}\varepsilon^7\varepsilon^{14}\varepsilon^{17}\varepsilon^7\varepsilon^2\varepsilon^7\varepsilon^{17}\right. \\ & \left.\varepsilon^8\varepsilon^6\varepsilon^{26}\varepsilon^{17}\varepsilon^6\varepsilon^4\varepsilon^{27}\varepsilon^{15}\varepsilon^9\varepsilon^4\varepsilon^{19}\varepsilon^{29}\varepsilon^{19}\varepsilon^7\varepsilon^9\right). \end{aligned}$$

Согласно теореме однозначности, для данной кодовой комбинации возможно провести децимации с индексами q_i равными: $q_i = 2^i$, $i = 1,2,3,4$.

Децимируем кодовую комбинацию $\{s\}$ со всеми индексами q_i и проведём процедуру декодирования согласно уравнениям (134). Полученные в результате декодирования комбинации запишем в таблицу 8.3.

Как видно из таблицы 8.3, декодирование комбинации с индексом децимации $q_i = 2^i$ приводит к циклическому сдвигу элементов декодированной комбинации на i шагов.

[ht]

Таблица 8.4

Децимирование кодовой последовательности $\{s\}$ для кода $(7,3)$ в случае пачки ошибок

Индекс децимации	Децимированная комбинация
$q = 2^0 = 1$	$\{u\}_1 = \{s\} = \begin{cases} \{s_0 & s_1 & s_2 & s_3 & s_4 & s_5 & s_6\} \\ \{s_0 & E & E & E & s_4 & s_5 & s_6\} \end{cases}$
$q = 2^1 = 2$	$\{u\}_2 = \begin{cases} \{u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6\} \\ \{s_0 & s_2 & s_4 & s_6 & s_1 & s_3 & s_5\} \\ \{s_0 & E & s_4 & s_6 & E & E & s_5\} \end{cases}$
$q = 2^2 = 4$	$\{u\}_4 = \begin{cases} \{u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6\} \\ \{s_0 & s_4 & s_1 & s_5 & s_2 & s_6 & s_3\} \\ \{s_0 & s_4 & E & s_5 & E & s_6 & E\} \end{cases}$

Необходимо отметить, что для каналов с пачками ошибок применение децимаций нерационально. Продемонстрируем подобную ситуацию на примере кодовой комбинации кода РСЭ $(7,3)$.

Кодовую комбинацию можно записать в общем виде, как

$$\{s\} = \{s_0 s_1 s_2 s_3 s_4 s_5 s_6\}.$$

Предположим, что на комбинацию $\{s\}$ была наложена пачка ошибок длиной 3:

$$\{s\} = \{s_0 E E E s_4 s_5 s_6\}.$$

Как можно видеть, при замыкании кодовой комбинации $\{s\}$ в кольцо можно выделить две k-элементные комбинации, которых может хватить для декодирования кодовой комбинации по принципу максимального правдоподобия. Теперь рассмотрим, что произойдёт, если к комбинации $\{s\}$ будут применены децимации. Эта ситуация представлена в таблице 8.4.

Можно видеть, что в обеих децимированных комбинациях не выделяется ни один безошибочный k-элементный участок. Таким образом, будут лишь появляться новые боковые «деревья» или увеличиваться высота существующих боковых «деревьев», что может привести как к отказу от декодирования, так и к выделению по мажоритарному принципу неверной информационной комбинации.

Таким образом, можно сделать вывод, что применение алгоритма МДБ с использованием децимаций для каналов с пачками ошибок нерационально.

8.6. Использование МДБ для обнаружения ошибок

Рассмотрим использование алгоритма МДБ для обнаружения ошибок.

Все k-элементные участки разрешённой последовательности $\{s\}$ при расчёте по формулам (132) и (7) [17, 43] дают одинаковый результат. Следова-

тельно, в случае появления в принятой кодовой комбинации ошибки, возникает несколько k-элементных участков, которые покажут результат, отличный от прочих k-элементных участков.

Таким образом, наличие ошибки в принятой кодовой комбинации определяется появлением хотя бы двух различных результатов, полученных при расчёте информационных элементов $A_1, A_2, \ldots, A_k$ по k-элементным участкам $\{s_i, s_{i+1}, \ldots, s_{i+k-1}\}$ принятой кодовой последовательности $\{s\}$ в соответствии с формулами (132) и (7) [17, 43].

Если рассмотреть данный процесс с точки зрения построения диаграммы «лес», то о наличии ошибки сигнализирует появление двух и более деревьев, как было показано на рис. 8.8.

Очевидно, что для обнаружения ошибки необходимо перебрать все n элементов принятой кодовой последовательности, а следовательно, достаточно перебрать $n-k+1$ k-элементный участок, не замыкая кодовую последовательность в кольцо.

Алгоритм обнаружения ошибок при использовании МДБ можно описать следующим образом:

1. Прием k первых символов кодовой последовательности.
2. Вычисление информационных элементов $A_1, A_2, \ldots, A_k$ по формуле (132).
3. Запись полученных значений в ячейки памяти.
4. Получение следующего символа кодовой последовательности, проведение вычислений и сравнение полученного результата с результатом, сохранённым на шаге (3).
5. В случае, если полученный результат совпадает с результатом, полученным при обработке k первых символов кодовой последовательности, то производится дальнейший приём кодовой комбинации; если же результаты не совпадают, то это обозначает, что в принятой комбинации содержится ошибка.

На рисунке 8.11 представлена блок-схема вышеприведенного алгоритма.

Далее на рисунке 8.12 приведём блок-схему устройства обнаружения ошибок, построенного на основе алгоритма МДБ.

Как следует из рассмотренной процедуры, в зависимости от места расположения ошибки для её определения необходимо получить от $k+1$ до n символов кодовой комбинации.

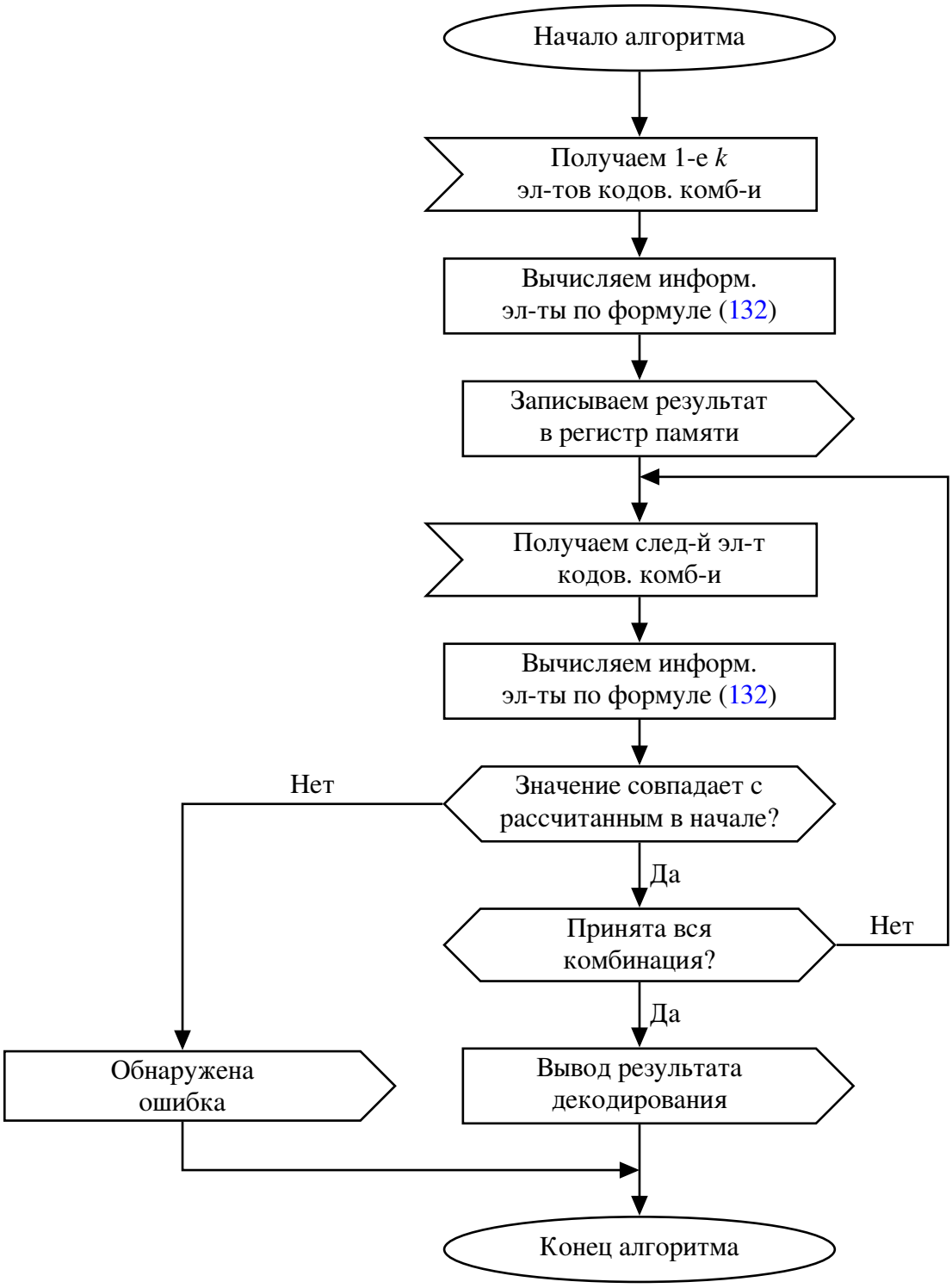

Рис. 8.11. Блок-схема алгоритма обнаружения ошибок в кодовой комбинации кода РСЭ на основе МДБ

8.7. Использование МДБ для декодирования кодовых комбинаций с ошибками

Рассмотрим применение алгоритма МДБ для исправления ошибок. Необходимо отметить, что в отличие от алгоритма обнаружения ошибок, при исправлении ошибок следует перебирать все n k-элементных участков. Так-

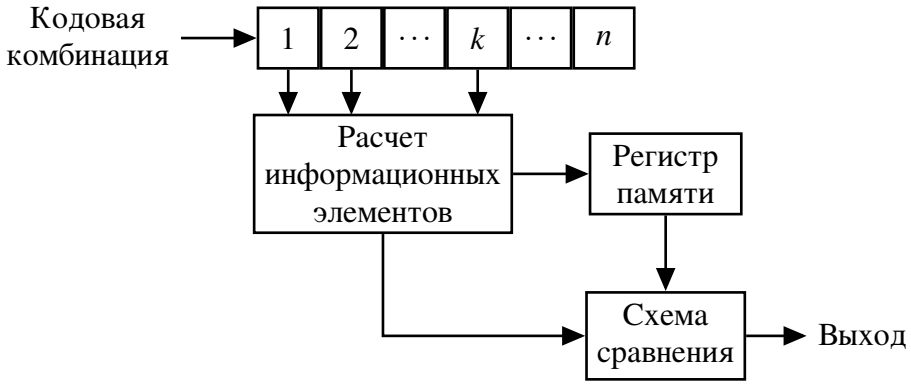

Рис. 8.12. Блок-схема декодирующего устройства кода РСЭ для обнаружения ошибок на основе двойственного базиса

же, в случае кодов РСЭ с сопряжёнными корнями возможно использование децимаций для увеличения исправляющей способности алгоритма.

Заметим, что поскольку алгоритм МДБ является мажоритарным, для принятия решения о правильности ответа достаточно, чтобы он был получен $\lfloor \frac{n}{2} \rfloor + 1$ раз, поскольку даже если все остальные k-элементные участки дадут совпадающий результат отличный от данного, его вес будем заведомо меньше.

Запишем алгоритм исправления ошибок при помощи МДБ без использования децимаций следующим образом:

1. Прием k первых символов кодовой последовательности.
2. Вычисление информационных элементов A_1, A_2, …, A_k по формуле (132).
3. Запись полученного результата в таблицу статистики.
4. Получение следующего символа кодовой последовательности, проведение вычислений и запись результата.
5. Декодирование завершается либо накоплением $\lfloor \frac{n}{2} \rfloor + 1$ одинакового результата, который и будет считаться однозначно соответствующим декодированной комбинации, либо получением всей кодовой комбинации. В последнем случае результатом декодирования будет либо результат с наибольшим весом, либо сигнал «отказ в декодировании», в случае появления в таблице статистики нескольких комбинаций с одинаковым весом, поскольку верную комбинацию при этом выделить невозможно. Сигнал «отказ в декодировании» может использоваться, например, в канале обратной связи, для запроса повторной передачи комбинации.

На рисунке 8.13 представлена блок-схема вышеприведенного алгоритма.

Как можно видеть из алгоритма и приведенной блок-схемы, декодирование принятой кодовой комбинации может завершиться еще до окончания ее приема.

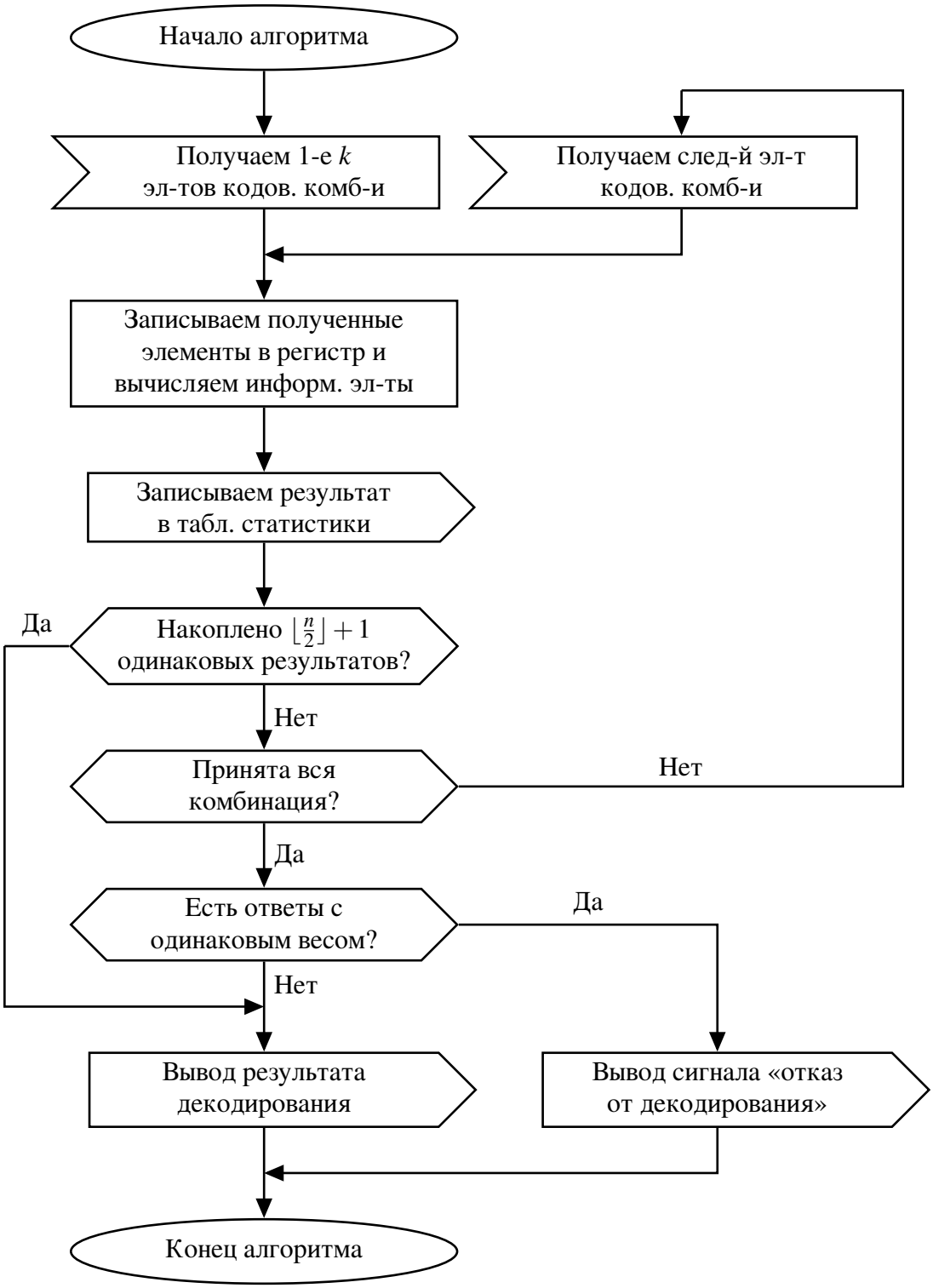

Рис. 8.13. Блок-схема алгоритма декодирования кода РСЭ на основе двойственного базиса

Далее, на рисунке 8.14, приведем блок-схему декодирующего устройства, реализующего вышеприведенный алгоритм. Устройство состоит из сдвигового циклического регистра на n ячеек, в который по мере получения записываются символы принимаемой кодовой комбинации; блока, осуществляющего расчет информационных элементов $A_1, A_2, \ldots, A_k$ по формуле (132);

блока накопления статистики и принятия решения, в котором хранятся результаты вычислений и осуществляется принятие решения о результате декодирования; также этот блок управляет ключевой схемой, расположенной на входе декодирующего устройства; ключевая схема блокирует вход сдвигового регистра на время обработки последних $k-1$ k-элементных комбинаций либо при досрочном декодировании кодовой комбинации до начала получения следующей кодовой комбинации.

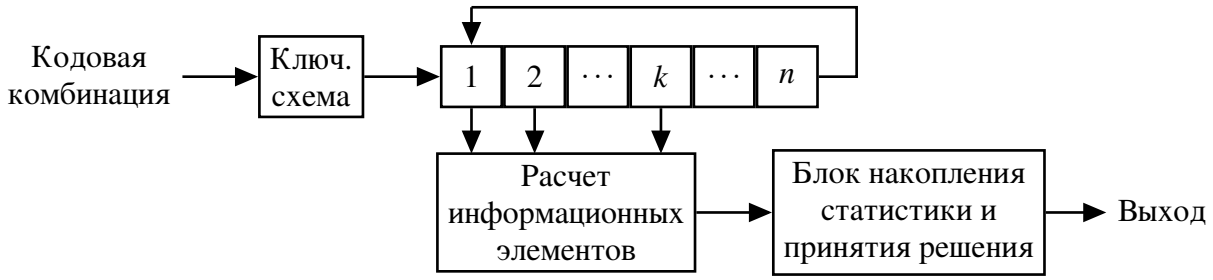

Рис. 8.14. Блок-схема декодирующего устройства кода РСЭ на основе двойственного базиса

Далее перейдём к рассмотрению того, как изменятся алгоритм и устройство декодирования кода РСЭ для случая использования децимаций.

Основным отличием станет добавление блока децимирования принятой комбинации, который будет либо сохранять децимированную комбинацию для дальнейшей обработки, либо передавать ее по мере перемешивания в соответствующие ветки устройства, которые параллельно будут обрабатывать все децимированные комбинации для увеличения скорости декодирования. Соответственно, в первом случае изменится часть устройства декодирования, отвечающая за вычисление информационных элементов, она должна будет корректировать результат вычислений согласно индексу децимации. Во втором случае, в устройстве добавятся параллельные ветки для обработки децимированных комбинаций, состоящие из регистра памяти и блока вычислений каждая. В первом варианте устройство усложнится незначительно, но декодирование будет занимать больше времени и досрочное декодирование станет невозможным. Второй вариант позволит сохранить время декодирования на том же уровне при усложнении конструкции.

На рисунке 8.15 приведена блок-схема устройства декодирования для первого варианта.

На рисунке 8.16 приведена блок-схема второго варианта декодера.

Рассмотрим процесс декодирования для каждого из вышеприведенных устройств.

Блок децимирования и хранения декодера, изображенного на схеме 8.15, по мере получения символов кодовой комбинации формирует децимиро-

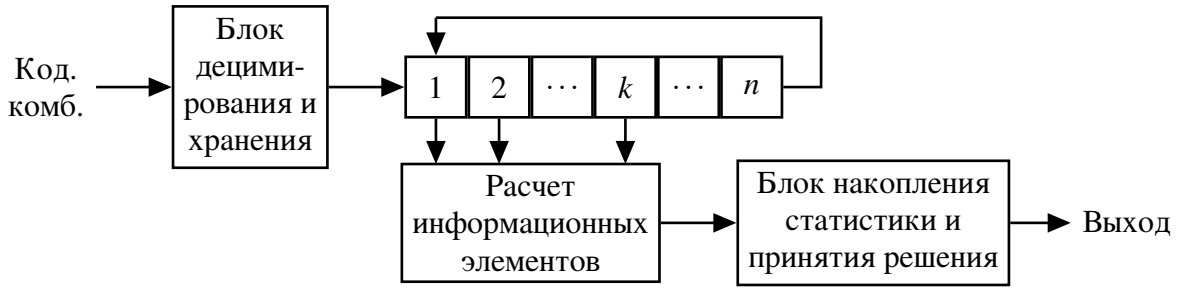

Рис. 8.15. Блок-схема декодирующего устройства кода РСЭ на основе двойственного базиса с использованием децимаций. Вариант 1

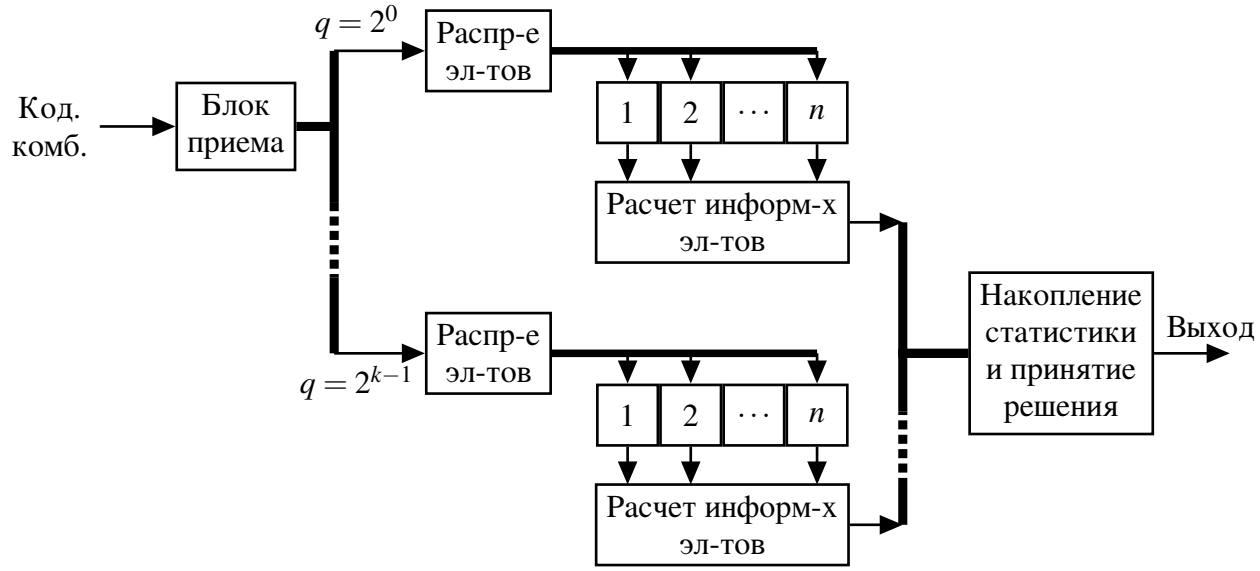

Рис. 8.16. Блок-схема декодирующего устройства кода РСЭ на основе двойственного базиса с использованием децимаций. Вариант 2

ванные комбинации и записывает их в память. После обработки пришедшей кодовой комбинации, в том случае, если определить информационные элементы не представляется возможным, блок децимирования и хранения блокирует прием (либо продолжает принимать последующие кодовые комбинации, записывая их в память) и начинает по очереди передавать на сдвиговый регистр децимированные комбинации. Таким образом, декодирование кодовой комбинации может как закончиться досрочно по накоплению $\lfloor \frac{n}{2} \rfloor + 1$ одинаковых результатов, так и продлиться до конца обработки децимированных комбинаций. В последнем случае время декодирования комбинации может превышать время приема кодовой комбинации.

Данную схему построения декодера можно применять в хороших каналах, где вероятность появления ошибок, требующих применения децимаций, мала.

При использовании декодера, изображенного на схеме 8.16, блок приема пересылает полученные символы кодовой комбинации в блоки распреде-

ления элементов, каждый из которых соответствует определенному индексу децимации. Блок распределения записывает полученные элементы в ячейки запоминающего регистра, соответственно своему индексу децимации. По мере появления в запоминающих регистрах k-элементных последовательностей, блоки расчета информационных элементов считывают их, производят расчет и пересылают результат в блок накопления статистики. Условие завершения декодирования то же, что и в первом случае, но данный декодер позволяет получить решение еще до приема всей кодовой комбинации, что достигается за счет распараллеливания процессов обработки децимированных комбинаций.

Данная схема декодера заметно сложнее в реализации поскольку требует в разы больше элементов (регистров памяти, сумматоров и умножителей) для построения. Такая схема может применяться в каналах с высокой вероятностью появления ошибок, когда известно, что большинство принятых комбинаций потребуют обработку при помощи децимаций.

8.8. Использование МДБ для декодирования кодовых комбинаций со стираниями

Теперь перейдём к рассмотрению использования алгоритма МДБ для исправления ошибок и стираний.

Первоначально рассмотрим само понятие стирания. Стирание представляет из себя символ принятой кодовой комбинации, значение которого невозможно определить точно. В случае двоичных сигналов, это обозначает невозможность определить принят сигнал, обозначающий единицу, или сигнал обозначающий ноль.

Для рассматриваемых в данной работе кодов РСЭ, стирание одного двоичного сигнала приводит к стиранию всего элемента кодовой комбинации, состоящего из m двоичных сигналов, где m — степень поля Галуа $\mathrm{GF}(2^m)$, на основе которого построен код.

При исправлении стираний можно использовать различные способы декодирования.

Один из способов заключается в том, чтобы подставить вместо стёртого символа все возможные его значения, декодировать все получившиеся варианты комбинации, а затем провести сравнение полученных результатов. Данный способ имеет ряд очевидных недостатков, основным из которых является необходимость проводить дополнительные операции декодирования, количество которых увеличивается с увеличением количества стираний.

При использовании мажоритарного алгоритма МДБ можно исключить стёртые символы из процесса декодирования, и отказаться от обработки k-элементных комбинаций, содержащих стёртые символы. В этом случае уменьшается количество требуемых расчётов, однако возникает следующая про-

[ht]

Таблица 8.5

Децимирование кодовой последовательности $\{s\}$ *со стираниями для кода* $(7,3)$

Индекс децимации	Децимированная последовательность
$q = 2^0 = 1$	$\{s_0 \quad X \quad s_2 \quad s_3 \quad X \quad s_5 \quad X\}$
$q = 2^1 = 2$	$\{s_0 \quad s_2 \quad X \quad X \quad X \quad s_3 \quad s_5\}$
$q = 2^2 = 4$	$\{s_0 \quad X \quad X \quad s_5 \quad s_2 \quad X \quad s_3\}$

блема. При определённом количестве стираний они могут расположиться таким образом, что невозможно будет выделить ни одного k-элементного участка. Например для кода $(7,3)$ это может выглядеть следующим образом:

$$\{s_0 X s_2 s_3 X s_5 X\},$$

здесь символом X обозначены стёртые позиции.

Видно, что в первом примере остаётся всего один возможный k-элементный участок, во втором примере таких участков уже нет. В этом случае возможно использование децимаций.

Рассмотрим применение децимаций при исправлении стираний. Для примера возьмём вторую последовательность с тремя стираниями из приведённого выше примера. Получившиеся децимированные последовательности сведём в таблицу 8.5.

Как можно видеть из таблицы 8.5, после проведения первой децимации ($q = 2$), мы получаем два k-элементных участка, которые позволят восстановить всю кодовую последовательность.

Таким образом, можно сделать вывод, что применение децимаций позволяет увеличить исправляющую способность алгоритма при исправлении стираний.

Далее рассмотрим алгоритм исправления стираний, с учётом использования децимаций. Учтём, что для начала использования децимаций не обязательно ждать приёма всей кодовой комбинации.

Вначале рассмотрим тривиальный случай, когда требуется лишь исправление стираний. Очевидно, что для этого достаточно одного k-элементного участка в любой из децимированных последовательностей. Данный алгоритм представлен на рисунке 8.17.

Далее рассмотрим вариант с обнаружением ошибок и исправлением стираний. Основным отличием от предыдущего варианта будет необходимость перебирать все элементы кодовой комбинации, проводить вычисления по всем обнаруженным k-элементным участкам, а затем проверять наличие

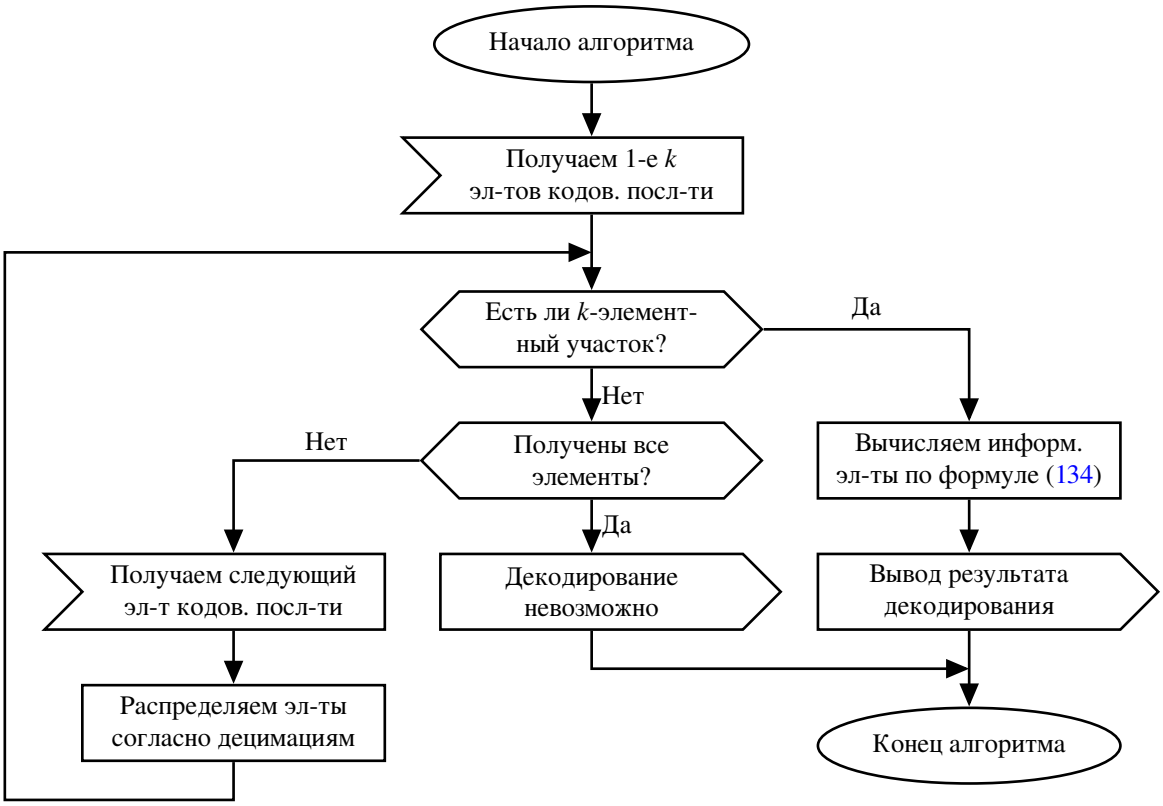

Рис. 8.17. Блок-схема алгоритма исправления стираний на основе двойственного базиса

ошибки, используя методику представленную в разделе 8.6. Блок-схема данного алгоритма представлена на рисунке 8.18.

По аналогии может быть построен алгоритм для исправления ошибок и стираний. При этом очевидно, что из-за того, что общее количество k-элементных участков уменьшится из-за наличия стираний, способность алгоритма к исправлению ошибок также уменьшится.

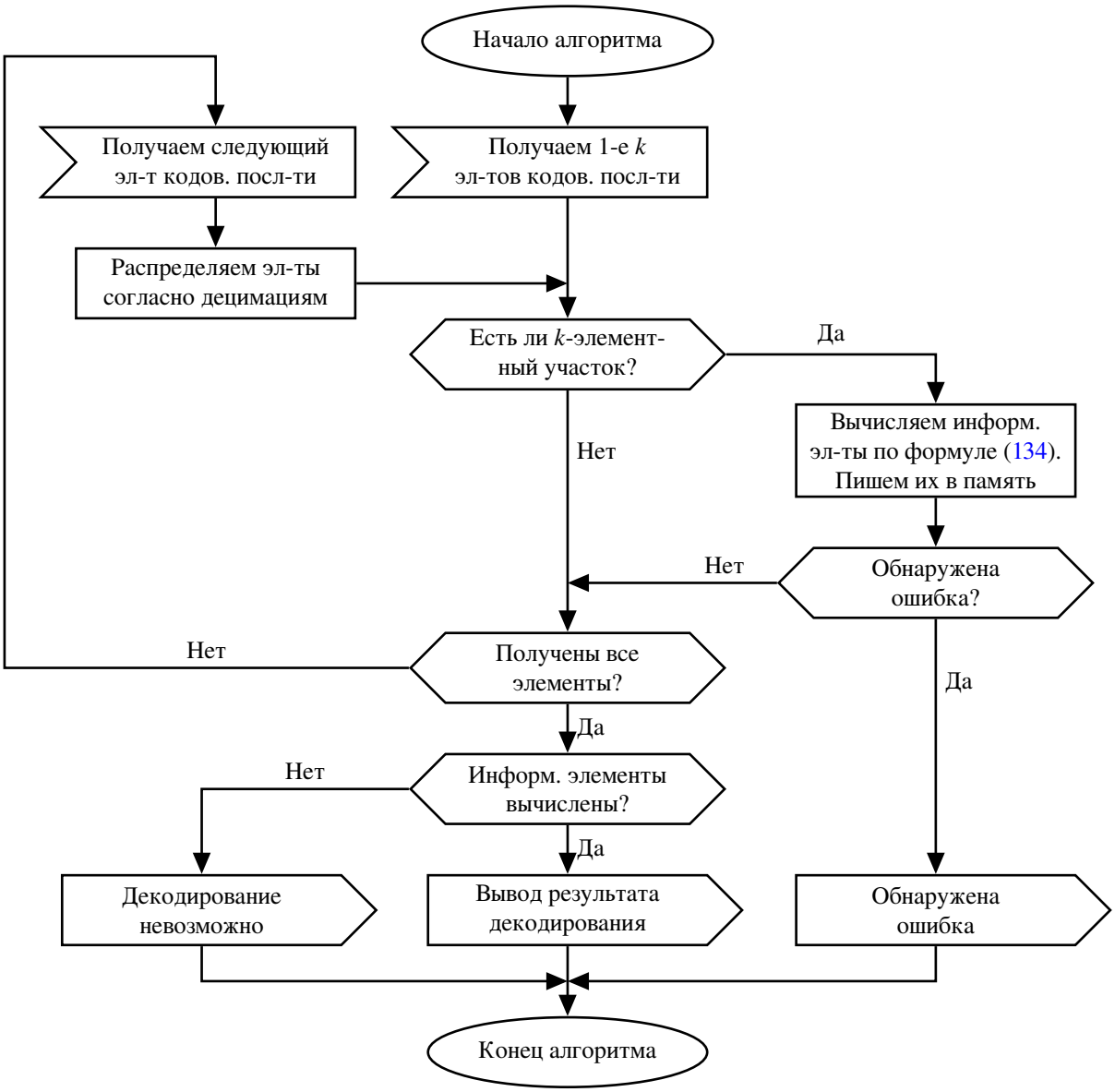

Рис. 8.18. Блок-схема алгоритма исправления стираний с обнаружением ошибок на основе двойственного базиса