

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

Кафедра радиотехнических систем

**С. Б. Саломатин, П. Г. Семашко**

## ***ИССЛЕДОВАНИЕ ЦИКЛИЧЕСКИХ КОДОВ***

Методическое пособие  
по курсам

«Теория кодирования и защита информации»,  
«Цифровая обработка сигналов и прикладная теория кодирования»  
для студентов радиотехнических специальностей  
всех форм обучения

Минск 2008

УДК 621. 391.25 (076)  
ББК 32.811 4 я 7  
С 16

**Р е ц е н з е н т**  
зав. кафедрой СиУТ БГУИР,  
д-р техн. наук, проф. В. К. Конопелько

**Саломатин, С. Б.**

**С 16** Исследование циклических кодов : метод. пособие по курсам «Теория кодирования и защита информации», «Цифровая обработка сигналов и прикладная теория кодирования» для студ. радиотех. спец. всех форм обуч. / С. Б. Саломатин, П. Г. Семашко. – Минск : БГУИР, 2008. – 32 с. : ил.  
ISBN 978-985-488-262-1

Пособие содержит теоретические сведения, алгоритмы, программы моделирования процессов кодирования и декодирования, а также исследования структурных свойств блочных циклических кодов. В лабораторной работе исследуются различные методы кодирования и декодирования на основе свойств синдрома ошибок (декодер Меггитта, метод вылавливания ошибок, мажоритарное декодирование). Процедуры кодирования и декодирования моделируются в программной среде Maple и среде моделирования Electronics Workbench.

**УДК 621. 391.25 (076)**  
**ББК 32.811 4 я 7**

**ISBN 978-985-488-262-1**

© Саломатин С. Б., Семашко П. Г., 2008  
© УО «Белорусский государственный  
университет информатики  
и радиозлектроники», 2008

## 1. ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ

1. Изучить методы помехоустойчивого кодирования и декодирования информации с помощью циклических блочных кодов.
2. Исследовать свойства циклических кодов и возможности применения свойств кодовых структур для кодирования и декодирования.
3. Приобрести навыки построения алгоритмов, программ и моделирующих структур для кодирования и декодирования информации циклическими кодами.

## 2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

### 2.1. Циклические коды

Циклические коды составляют класс кодов, исправляющих и контролируемых ошибки, кодирование и декодирование которых основано на полиномиальном представлении.

*Определение.* Линейный  $(n, k)$ -код  $C$  над полем  $F_q$  называется циклическим, если из того, что вектор  $(c_0, c_1, \dots, c_{n-1})$  принадлежит  $C$ , следует, что его циклический сдвиг  $(c_{n-1}, c_0, c_1, \dots, c_{n-2})$  принадлежит  $C$ .

Линейный код над полем  $GF(q)$  длиной  $n$  представляет собой подпространство пространства  $GF(q^n)$ . Каждый вектор из этого пространства можно представить многочленом от  $x$  степени не выше  $n - 1$ . Компоненты вектора отождествляются с коэффициентами многочлена.

Сопоставим кодовому слову  $c$  полином  $c(x)$ :

$$c = (c_0, c_1, \dots, c_{n-1}) \rightarrow c(x) = c_0 + c_1 x + \dots + c_{n-1} x^{n-1}.$$

Переменная  $x$  служит индикатором относительно положения элемента  $c_j$  в виде терма (монома)  $c_j x^j$  полинома  $c(x)$ . В полиномиальном представлении циклический сдвиг на одну позицию  $D_c \{ c(x) \}$  соответствует умножению на  $x$  по модулю  $(x^n - 1)$ :

$$D_c \{ c(x) \} = x c(x) \bmod (x^n - 1), \quad (1)$$

где  $D_c$  – оператор циклического сдвига.

Операция циклического сдвига реализуется на *регистре сдвига*. Программная функция циклического сдвига на  $n$ -разрядном регистре сдвига имеет вид

$$f := (c(n), c(n-1), \dots, c_2, c_1) \rightarrow (c_1, c(n), c(n-1), \dots, c_3, c_2);$$

Циклический сдвиг может быть записан через умножение в кольце многочленов  $GF(q)[x]/(x^n - 1)$ .

$$x p(x) = \text{rem}\{ (x p(x)) / (x^n - 1) \}, \quad (2)$$

где  $\text{rem}\{b(x)/z(x)\}$  – остаток от деления полиномов  $b(x)$  на  $z(x)$ .

*Корни многочлена и минимальные функции.* Элемент  $b \in GF(q^n)$  называется корнем многочлена  $a(x)$  над  $GF(q^a)$ , если  $a(b) = 0$ . Отсюда следует, что многочлен  $a(x)$  делится без остатка на многочлен  $x + b$ , т. е.  $a(x) = (x + b)q(x)$ .

*Неприводимым многочленом* называется такой многочлен степени  $p$  над полем  $GF(q^n)$ , который не делится ни на один из многочленов степени, меньшей  $p$  (но не больше нуля), с коэффициентами из того же поля.

*Минимальной функцией*, или *минимальным многочленом*, над  $GF(q^a)$  для элемента  $b \in GF(q^n)$  называется нормированный многочлен  $m(x)$  наименьшей степени с коэффициентами из  $GF(q^a)$ , не приводимый над этим полем, такой, что  $m(b) = 0$ .

*Порождающий и проверочный полиномы.* Важным свойством циклических кодов является то, что все кодовые слова-полиномы кратны одному фиксированному полиному  $g(x)$ , который называется *порождающим полиномом кода*. Порождающий полином  $g(x)$  является делителем бинома  $(x^n - 1)$ .

Множество  $\{x^i g(x); 0 \leq i \leq k - 1\}$  можно рассматривать как базис размерностью  $k$  циклического кода  $C$ .

Генераторная матрица кода представляется как матрица размером  $(k \times n)$  следующего вида:

$$\mathbf{G} = \begin{bmatrix} g(x) \\ x \cdot g(x) \\ x^2 \cdot g(x) \\ \mathbf{L} \mathbf{L} \mathbf{L} \\ x^{k-1} \cdot g(x) \end{bmatrix} = \begin{bmatrix} g_0 & g_1 & \dots & g_{n-k} & 0 & \dots & 0 \\ 0 & g_0 & \dots & \dots & g_{n-k} & \dots & 0 \\ 0 & 0 & g_0 & g_1 & \dots & g_{n-k} & \dots \\ \mathbf{L} & \mathbf{L} & \mathbf{L} & \mathbf{L} & \mathbf{L} & \mathbf{L} & \mathbf{L} \\ 0 & 0 & 0 & \dots & g_0 & \dots & g_{n-k} \end{bmatrix}. \quad (3)$$

Строки матрицы  $\mathbf{G}$  линейно независимы, и, таким образом, ранг матрицы  $\mathbf{G}$  равен  $k$  – размерности кода  $C$ . С помощью такой матрицы осуществляется несистематическое кодирование.

*Пример.* Циклический код Хэмминга (7, 4, 3) задается порождающим полиномом  $g(x) = x^3 + x + 1$ , имеющим двоичное представление в виде кода (1101). Построить порождающую матрицу несистематического кода.

*Решение.* Воспользуемся схемой построения (3). Получим матрицу следующего вида:

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

Если проверочная часть порождающей матрицы строится с помощью полиномов  $x^{n-1} \bmod g(x)$ , ...,  $x^{n-k-1} \bmod g(x)$ , то в этом случае реализуется систематическое кодирование.

*Пример.* Циклический код Хэмминга (7, 4, 3) задается порождающим полиномом  $g(x) = x^3 + x + 1$ , имеющим двоичное представление в виде кода (1101). Построить порождающую матрицу систематического кода.

*Решение.* Обратимся к полиномам для систематического кодирования:

$$x^m \bmod g(x), m = n - 1, \dots, n - k - 1.$$

Процесс вычисления может быть выполнен с помощью программы  
 $\text{> rem}(x^m, x^3 + x + 1, x) \bmod 2$ ;

Тогда имеем

$$x^6 \bmod (x^3 + x + 1) = x^2 + 1,$$

$$x^5 \bmod (x^3 + x + 1) = x^2 + x + 1,$$

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

$$x^3 \bmod (x^3 + x + 1) = x + 1.$$

Следовательно, систематическая порождающая матрица кода имеет вид

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

Порождающий полином должен удовлетворять следующим требованиям:

- 1)  $g(x)$  должен быть ненулевым;
- 2) вес  $g(x)$  не должен быть меньше минимального кодового расстояния:  
 $wt(g(x)) \geq d_{\min}$ ;
- 3)  $g(x)$  должен иметь максимальную степень  $(n - k)$ , которая определяет число избыточных элементов в коде;
- 4)  $g(x)$  должен быть делителем полинома  $(x^n - 1)$ .

Для того чтобы найти некоторый порождающий полином, надо знать разложение бинома  $(x^n - 1)$  на неприводимые множители  $m_j(x)$ ,  $j = 1, 2, \dots, l$ :

$$(x^n - 1) = m_1(x) m_2(x) \dots m_l(x).$$

Представим многочлен  $(x^n - 1)$  как произведение минимальных многочленов:

$$x^n - 1 = \prod_{i \in J} m_i(x),$$

где  $J$  – множество индексов. Разобьем это множество на два непересекающихся подмножества –  $J_g$  и  $J_h$  – так, что  $J = J_g \cup J_h$ . Введем два полинома:

$$g(x) = \prod_{i \in J_g} m_i(x) = g_0 + g_1x + g_2x^2 + \dots + g_rx^r,$$

$$h(x) = \prod_{i \in J_h} m_i(x) = h_0 + h_1x + h_2x^2 + \dots + h_kx^k.$$

Многочлены  $g(x)$  и  $h(x)$  удовлетворяют условию  $g(x)h(x) = (x^n - 1)$ . Если полином  $g(x)$  определим как порождающий, то любое кодовое слово можно представить в виде  $c(x) = a(x)g(x)$ , где  $a(x)$  – информационный полином. Следовательно, имеет место равенство

$$c(x)h(x) = a(x)g(x)h(x) = a(x)(x^n - 1).$$

Откуда следует, что

$$c(x)h(x) \equiv 0 \pmod{(x^n - 1)}.$$

Многочлен  $h(x)$  называется *проверочным полиномом* и вычисляется как

$$h(x) = \frac{x^n - 1}{g(x)}. \quad (4)$$

Проверочная матрица циклического кода  $C$  имеет вид

$$H = \begin{bmatrix} 0 & 0 & K & 0 & h_k & h_{k-1} & K & h_0 \\ 0 & 0 & K & 0 & h_k & h_{k-1} & K & h_0 & 0 \\ & & K & & & & K & & \\ h_k & h_{k-1} & K & h_0 & 0 & & K & 0 \end{bmatrix} \quad (5)$$

или

$$H = \begin{bmatrix} h_k & h_{k-1} & K & h_0 & 0 & 0 & K & 0 & 0 \\ 0 & h_k & K & h_2 & h_1 & h_0 & 0 & K & 0 & 0 \\ & & K & & & & K & & & \\ 0 & 0 & K & 0 & h_k & h_{k-1} & K & h_1 & h_0 \end{bmatrix}. \quad (6)$$

Код с порождающей матрицей  $H$ , т.е. код, дуальный к коду  $C$ , также является циклическим кодом. Кодовые слова по полиному  $h(x)$  образуют нуль-пространство относительно кодовых слов, построенных по полиному  $g(x)$ .

*Соотношение степеней.* Для циклического кода число информационных символов  $k$  и число проверочных символов  $r$  определяется как

$$k = \deg h(x), \quad r = \deg g(x). \quad (7)$$

*Пример.* Построить циклический код над  $GF(2)$  длиной  $n = 31$  с числом проверочных символов  $r = 16$ .

*Решение.* Проведем факторизацию (разложение на множители) полинома  $x^{31} - 1$ . Для этого воспользуемся оператором *Factor* в среде Maple:

$$> \text{Factor}((x^{31} - 1)) \pmod{2};$$

Получим следующий результат:

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

Введем следующие обозначения:

$$m_0(x) = x + 1; \quad m_1(x) = x^5 + x^2 + 1; \quad m_3(x) = x^5 + x^4 + x^3 + x^2 + x + 1;$$

$$m_5(x) = x^5 + x^4 + x^2 + x + 1; \quad m_7(x) = x^5 + x^3 + x^2 + x + 1;$$

$$m_{11}(x) = x^5 + x^4 + x^3 + x + 1; \quad m_{13}(x) = x^5 + x^3 + 1.$$

Можно убедиться, что данные полиномы являются минимальными функциями  $m_{a^i}(x) = m_i(x)$  для элементов  $a^i$  поля  $GF(2^5)[x]/(x^5 + x^2 + 1)$ . Для этого достаточно воспользоваться программой «MinPoly» из библиотеки вычислений «Циклические коды».

Таким образом, разложение многочлена  $x^{31} - 1$  задается следующим образом:

$$x^{31} - 1 = m_0(x)m_1(x)m_3(x)m_5(x)m_7(x)m_{11}(x)m_{15}(x) \bmod 2.$$

Множество индексов  $J$  имеет вид  $J = \{0, 1, 3, 5, 7, 11, 15\}$ .

Для  $r = 16$  выберем полином

$$g(x) = m_0(x)m_1(x)m_3(x)m_5(x) = x^{16} + x^{15} + x^{12} + x^7 + x^6 + x^5 + x^4 + 1.$$

Для  $k = 15$  построим полином

$$h(x) = m_7(x)m_{11}(x)m_{15}(x) = x^{15} + x^{14} + x^{13} + x^{12} + x^{10} + x^8 + x^7 + x^6 + x^5 + x^4 + 1.$$

## 2.2. Кодирование циклических кодов

1. Кодирование с помощью порождающего полинома  $g(x)$ .

Кодирование циклического кода может быть *систематическим* (разделимым) или *несистематическим* в зависимости оттого, что именно делается с сообщением:

- несистематическое кодирование:

$$c(x) = a(x)g(x); \quad (8)$$

- систематическое кодирование:

$$c(x) = x^{n-k}a(x) + \text{rem}\{x^{n-k}a(x)/g(x)\}, \quad (9)$$

где  $a(x) = a_0 + a_1x + \dots + a_{k-1}x^{k-1}$  – полином информационного сообщения.

При систематическом кодировании полином кодового слова  $c(x)$  находится с помощью соотношения

$$c(x) = a_u(x) + b(x),$$

где  $a_u(x) = a_0x^{n-k} + a_1x^{n-k+1} + \dots + a_{k-1}x^{n-1}$  – многочлен информационных символов;  $b(x) = c_kx^{r-1} + c_{k+1}x^{r-2} + \dots + c_{n-2}x + c_{n-1}$  – многочлен проверочных символов.

Таким образом, для нахождения многочлена проверочных символов при систематическом кодировании достаточно разделить многочлен информационных символов на порождающий полином. Остаток от деления и будет искомым многочленом проверочных символов.

*Пример.* Рассмотрим циклический код Хэмминга (7, 4, 3) с порождающим полиномом  $g(x) = x^3 + x + 1$ . Зададим информационный вектор в виде четырех двоичных символов:  $\mathbf{a} = (0, 1, 1, 0)$ . Требуется получить вектор  $\mathbf{c}$  систематического кода.

*Решение.* Проведем следующие вспомогательные вычисления в среде Maple. Переведем вектор  $\mathbf{a} = (0, 1, 1, 0)$  в полином  $a(x)$ . Для этого определим вектор как множество:

$> a := \text{array}([0, 1, 1, 0], 1..4):$

Используя оператор

$> \text{ToPoly}(a, 2, x);$

получим результат перевода вектора в полином  $a(x) = x^2 + x \rightarrow a := x + x^2$ .

Вычислим полином кодового слова по формуле (9) для систематического кодирования :

$> c := \text{sort}(\text{simplify}(a * x^3 + \text{rem}(x^3 * a, (x^3 + x + 1), x) \bmod 2));$

Получим

$$c := x^5 + x^4 + 1 \rightarrow c(x) = x^5 + x^4 + 1.$$

Для представления результата в виде вектора воспользуемся оператором

$> c := \text{ToVector}(c, x, 7, 2);$

Получим окончательный результат:

$$c := [1, 0, 0, 0, 1, 1, 0].$$

2. *Кодирование циклическим кодом с помощью проверочного многочлена  $h(x)$ .*

Систематическое кодирование низкоскоростным циклическим кодом удобно проводить с помощью проверочного полинома, имеющего в данном случае меньшую размерность, чем порождающий многочлен.

Коэффициенты многочлена проверочных символов  $b(x)$  вычисляются по коэффициентам многочленов  $a(x)$  и  $h(x)$  следующим образом:

$$c_k = \sum_{j=0}^{k-1} h_j a_{i+j}, \quad i = 0, 1, \dots, n - k - 1.$$

Справедливость данной формулы вытекает из равенства  $\mathbf{cH}^T = 0$ . Эти соотношения позволяют по заданным информационным символам вычислить проверочный символ  $c_k$ , затем, зная  $c_k$ , вычислить  $c_{k+1}$  и т. д.

Схемы кодеров, построенные на регистрах сдвига и логических схемах, по рассмотренным выше алгоритмам, приведены на рис. 1–3.

3. *Кодирование циклическим кодом путем задания корней всех кодовых полиномов.*

Кодирование предполагает переход в соответствующее расширенное поле  $GF(p^m)$ . Условие, что все кодовые полиномы делятся на порождающий многочлен  $g(x)$ , означает, что все полиномы слов кода должны принимать нулевое значение на корнях полинома  $g(x)$ .



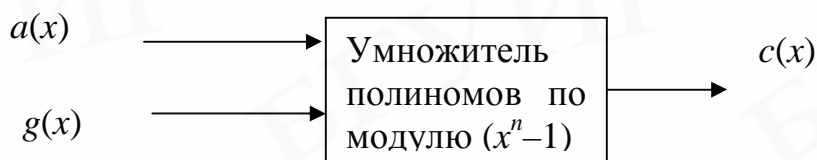


Рис. 1. Кодер несистематического кода

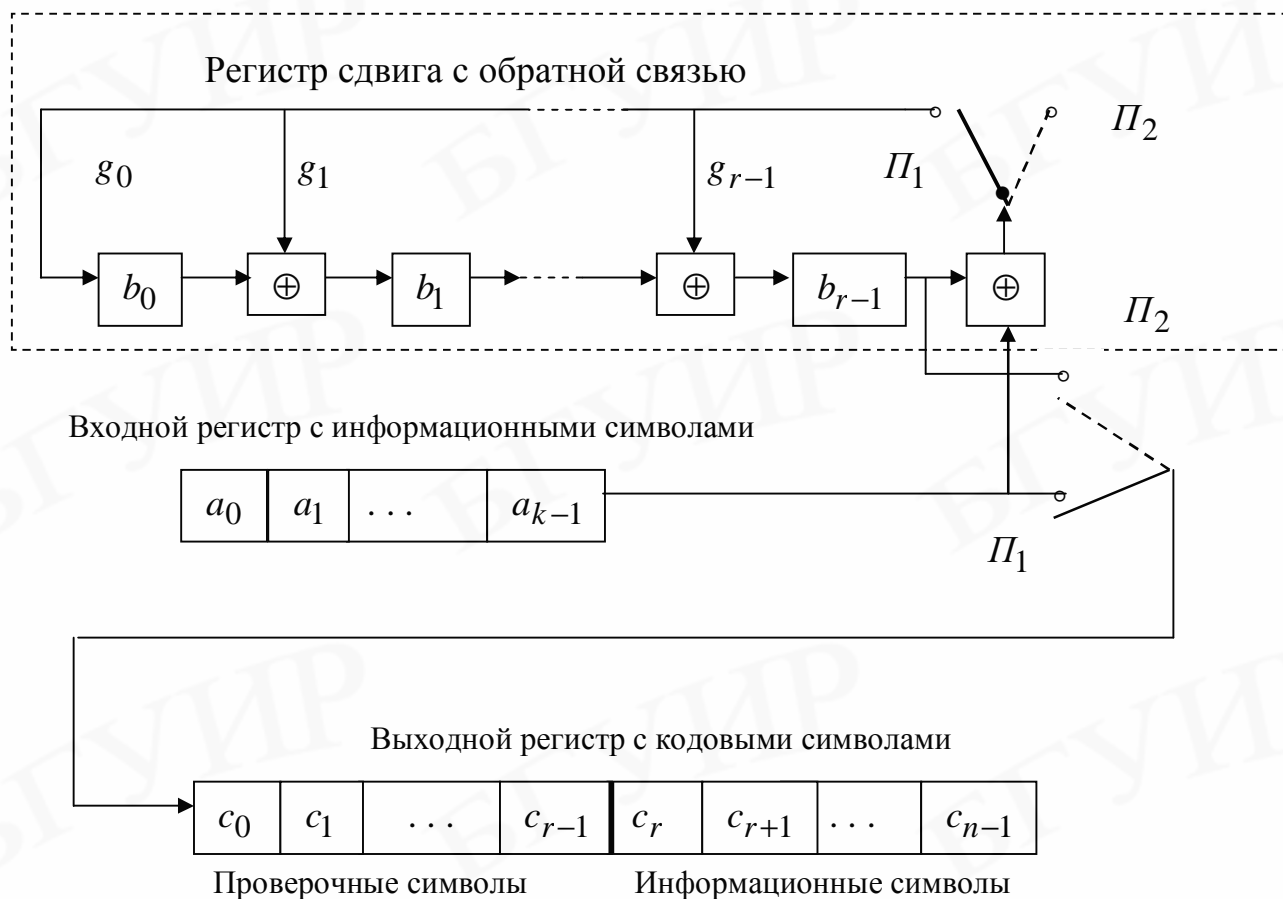


Рис. 2. Кодер систематического кода, задаваемый  $g(x)$

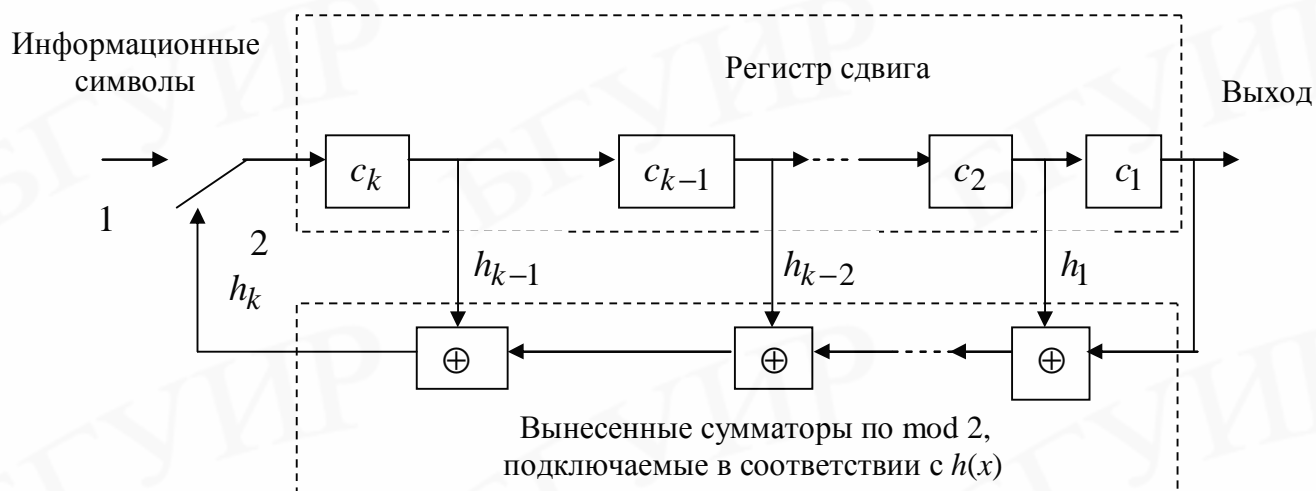


Рис. 3. Кодер систематического кода, задаваемый  $h(x)$

Пусть  $C \subseteq GF(q)[x]/(x^n - 1)$  – циклический код с порождающим полиномом  $g(x)$ , и пусть  $\{b_1, \dots, b_{n-k}\}$  – корни полинома  $g(x)$ . Полином  $c(x)$  является кодовым полиномом тогда и только тогда, когда вектор  $(c_0, c_1, \dots, c_{n-1})$ , образованный коэффициентами полинома  $c(x)$ , лежит в нуль-пространстве матрицы:

$$H = \begin{bmatrix} 1 & b_1 & b_1^2 & \dots & b_1^{n-1} \\ \mathbf{M} & & & & \\ 1 & b_{n-k} & b_{n-k}^2 & \dots & b_{n-k}^{n-1} \end{bmatrix}. \quad (10)$$

*Пример.* Построить проверочную матрицу циклического кода с параметрами (15, 11), используя метод задания корней кодовых слов.

*Решение.* Определим полином  $f = x^{15} - 1$ . Разложим данный полином на множители, применив оператор:

```
> Factor((x^15 - 1)) mod 2;
(x^4 + x + 1)*(x^2 + x + 1)*(x^4 + x^3 + x^2 + x + 1)*(x^4 + x^3 + 1)*(x + 1) =
= (x^4 + x + 1)(x^2 + x + 1)(x^4 + x^3 + x^2 + x + 1)(x^4 + x^3 + 1)(x + 1).
```

Выберем из полученного результата полином  $g(x) = (x^4 + x + 1)$ . Используя оператор `irreduc(x^4 + x + 1)`, убедимся, что данный полином неприводимый и может служить в качестве полинома для построения поля  $GF(2^4)$ .

Определим примитивный элемент поля

```
> alias (alpha = RootOf(Z^4 + Z + 1) mod 2);
```

и построим элементы поля

```
> for i to 2^4-2 do powers[i] := evala (alpha^(i - 1)) mod p; od;
1, a, a^2, a^3, 1+a, a+a^2, a^2+a^3, 1+a+a^3, 1+a^2, a+a^3, 1+a+a^2,
a+a^2+a^3, 1+a+a^2+a^3, 1+a^2+a^3+1+a^3.
```

Найдем корни полинома  $(x^4 + x + 1)$  в расширенном поле путем разложения полинома на элементарные множители:

```
> Factor(g, alpha) mod 2;
(x + alpha + 1)*(x + alpha)*(x + alpha^2 + 1)*(x + alpha^2)
```

Таким образом, корнями полинома  $(x^4 + x + 1)$  являются следующие элементы поля:

$$b_1 = a; \quad b_2 = a^2; \quad b_3 = 1 + a = a^4; \quad b_4 = 1 + a^2 = a^8.$$

Используя выражение (10) и представляя корни  $\beta$  как векторы-столбцы, после соответствующих упрощений получим

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \end{bmatrix}.$$

Матрица **Н** является проверочной матрицей эквивалентного кода Хэмминга.

### 2.3. Алгоритмы вычисления остатка от деления двух полиномов

Остаток можно вычислить с помощью регистра сдвига, обратные связи которого определяются делителем (рис. 4). Если на вход подавать делимое, то частное будет появляться в виде последовательности символов в цепи обратной связи, а остаток содержаться в регистре.

*Пример.* Составить программу и вычислить остаток от деления полинома  $bp(x) = x^8 a(x) = x^8(x^6 + x^4 + x^3 + x^2 + 1)$  на порождающий полином  $g(x) = x^8 + x^7 + x^6 + x^4 + 1$  кода (15, 7, 5) по схеме регистра сдвига с обратной связью.

*Решение.* Зададим коэффициенты полинома  $g(x)$ :

> g0:=1; g1:=0; g2:=0; g3:=0; g4:=1; g5:=0; g6:=1; g7:=1; g8:=1;

Запишем функцию обратной связи регистра делителя в виде

> f := (r8, c7, r6, r5, r4, r3, r2, r1) -> (r1 + s mod 2, (r8 + g1\*r1 mod 2), (r7 + g2\*r1 mod 2), (r6 + g3\*r1 mod 2), (r5 + g4\*r1 mod 2), (r4 + g5\*r1 mod 2), (r3 + g6\*r1 mod 2), (r2 + g7\*r1 mod 2));

Зададим делимое в виде полинома  $bp(x) = x^{14} + x^{12} + x^{11} + x^{10} + x^8$  и в виде вектора из 15 символов  $bp(x) \rightarrow b := [0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1]$ ;

Процесс вычисления остатка представим в виде программы

> for i from 1 to 15 do

s := b[15 - i + 1]; print(i); r := f(r); od;

которая позволяет контролировать все состояния регистра в процессе вычисления. Результат вычисления (остаток), получаемый на 15-м шаге, равен

$r := 0, 0, 0, 1, 1, 1, 1, 1 \rightarrow x^7 + x^6 + x^5 + x^4 + x^3$ .

Проверим результат, проводя вычисления в полиномиальной форме:

> r := sort (expand (rem (bp, g, x) mod 2));

$r := x^7 + x^6 + x^5 + x^4 + x^3$ .

Схемы кодеров систематического кода используют ввод делимого в цепь обратной связи другим способом (см. рис. 2), который позволяет сократить число шагов вычислений. Функция обратной связи в этом случае запишется как

$f := (r(n - k), c(n - k - 1), \dots, r2, r1) \rightarrow (w * g0, (r8 + g1 * w \text{ mod } 2), (r7 + g2 * w \text{ mod } 2), (r6 + g3 * w \text{ mod } 2), (r5 + g4 * w \text{ mod } 2), (r4 + g5 * w \text{ mod } 2), (r3 + g6 * w \text{ mod } 2), (r2 + g7 * w \text{ mod } 2));$

Цикл вычисления остатка примет вид

> for i from 1 to k do

s := b[15 - i + 1]; w := s + op(n - k, [r]) mod 2; (print (i); r := f(r); od;

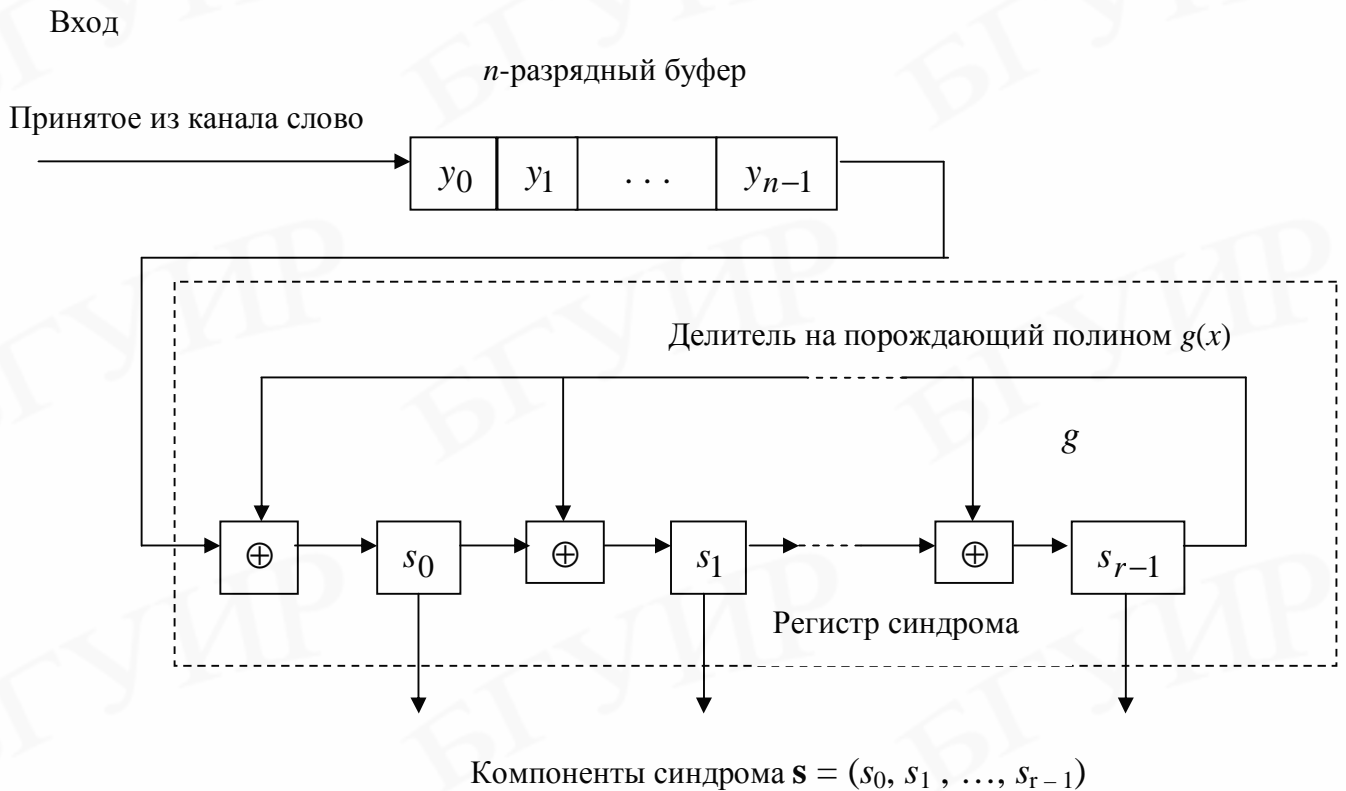


Рис. 4. Схема вычисления синдрома

## 2.4. Синдромные полиномы

Основным этапом большинства алгоритмов декодирования является вычисление синдрома принятой последовательности. Синдром – это вектор длиной  $n-k$ , равный произведению принятой последовательности на проверочную матрицу. В случае полиномиальных циклических кодов для синдрома имеется три тесно связанные друг с другом записи.

1. Синдром можно определить как многочлен  $s(x)$  степени  $(n - k)$ , являющийся остатком от деления принятой последовательности  $\mathbf{y} \rightarrow y(x)$  на порождающий многочлен (см. рис. 4):

$$s(x) \equiv \text{rem}\{y(x)/g(x)\} \bmod g(x).$$

Поскольку все кодовые слова делятся на  $g(x)$ , то  $s(x)$  не зависит от переданного слова, а зависит лишь от комбинации ошибок.

2. Второй вид синдрома может быть введен в случае, если порождающий многочлен является произведением двух или большего числа неприводимых сомножителей. Предположим, что

$$g(x) = g_1(x)g_2(x)\dots g_j(x).$$

Тогда синдром можно представить в виде набора многочленов:

$$\begin{aligned} s_1(x) &\equiv y(x) \bmod g_1(x), \\ s_2(x) &\equiv y(x) \bmod g_2(x), \\ &\dots\dots\dots \\ s_j(x) &\equiv y(x) \bmod g_j(x). \end{aligned}$$

Набор синдромов и синдром  $s(x)$  несут одну и ту же информацию, что следует из китайской теоремы об остатках многочленов [ 1 ].

3. Третья форма синдрома состоит в том, чтобы определить векторы форм синдромов  $s_1, s_2, \dots$  в виде

$$\mathbf{s}_1 = y(\mathbf{b}_1), \quad \mathbf{s}_2 = y(\mathbf{b}_2), \mathbf{K},$$

где  $b_1$  – корень  $g_1(x)$ ;  $b_2$  – корень  $g_2(x)$  и т. д.

Эта форма синдрома соответствует проверочной матрице, задаваемой корнями.

### 2.4.1. Пакеты ошибок

Характерной особенностью циклических кодов является способность к распознаванию пакетов ошибок. Под пакетом ошибок понимается группирование ошибок в одной ограниченной области кодового слова. Пакет ошибок в полиномиальном виде можно представить как

$$e(x) = x^j b(x).$$

Например, задавая  $b(x)=1+x+x^3+x^6$ , пакет ошибок в векторном виде будет иметь вид

$$\mathbf{c} = ( \begin{smallmatrix} 0\mathbf{K}0 \\ \text{Сегмент} \\ \text{из } j \text{ нулей} \end{smallmatrix} \quad \begin{smallmatrix} 1101001 \\ \text{Пакет} \\ x^j b(x) \end{smallmatrix} \quad 0\mathbf{K}0 ).$$

Пакет ошибок начинается и заканчивается отличным от нуля символом. Если длина пакета не превосходит величину  $r = n - k$ , то степень полинома ошибок меньше  $r$ . В этом случае  $e(x)$  не делится на  $g(x)$  без остатка и синдром принятого слова всегда отличен от нулевого. Пакет ошибок длиной, равной или меньшей  $r$ , всегда распознается. Распознается также любой циклический сдвиг многочлена  $b(x)$  степени, меньшей  $r$ . Для циклического  $(n, k)$ -кода доля необнаруживаемых пакетов ошибок длины  $l > r + 1$  равна  $2^{-r}$ .

## 2.5. Методы декодирования циклических кодов

### 2.5.1. Декодеры Меггитта

В основе декодера Меггитта лежат следующие свойства циклических кодов:

1. Существует взаимно однозначное соответствие между множеством всех исправляемых ошибок и множеством всех синдромов.
2. Если  $s(x)$  – синдром, соответствующий многочлену ошибок  $e(x)$ , то  $xs(x) \bmod g(x)$  – синдром, соответствующий  $xe(x) \bmod (x^n - 1)$ .

3. Пусть  $s(x)$  – синдром принятого из канала слова  $y(x)$  некоторого циклического кода  $(n, k)$ . Обозначим через  $s_1(x) = \text{rem}\{ (xs(x)) / g(x) \}$  – остаток от деления многочлена  $xs(x)$  на порождающий полином  $g(x)$ . Тогда  $s_1(x) = s^{(1)}(x)$  является синдромом циклического сдвига принимаемого слова  $y^{(1)}(x) = D\{y(x)\}$ , т.е. остатком от деления циклического сдвига  $y^{(1)}(x)$  на порождающий полином  $g(x)$ .

Из вышеприведенных свойств вытекает:

- если комбинация ошибок циклически сдвинута на одну позицию вправо, то для получения нового синдрома для  $y^{(1)}(x)$  следует сдвинуть содержимое регистра сдвига с обратными связями, содержащего  $s(x)$ , на одну позицию вправо;
- множество всех комбинаций ошибок можно разбить на классы эквивалентности таким образом, чтобы каждый класс состоял из циклических сдвигов одной комбинации ошибок.

Если длина циклического кода равна  $n$ , то для определения того, принадлежит ли комбинация ошибок данному классу, нужно сдвинуть регистр с обратными связями, содержащий  $s(x)$ , не более  $n$  раз и сравнить содержимое регистра с фиксированной последовательностью после каждого шага.

Схема декодера Меггитта приведена на рис. 5. Перед началом работы регистры сдвига декодера обнуляются. На первых  $n$  тактах принятое слово загружается в верхний регистр, а в нижнем регистре вычисляется синдром многочлена  $y(x)$ . После этого вход декодера отключается и в течение последующих  $n$  тактов проводится покомпонентное обнаружение и исправление ошибок. В каждом такте с помощью логической схемы распознавания ошибок проверяется соответствие текущего синдрома ошибочной  $(n - 1)$ -й компоненте сдвинутого многочлена  $y(x)$ . При обнаружении такого соответствия на следующем такте декодирования ошибка исправляется, а синдром модифицируется.

#### 2.5.2. Перестановочное декодирование и метод вылавливания ошибок

Предположим, что требуется исправить все векторы ошибок

$$\mathbf{e} = (e_0, \dots, e_{v-1}),$$

вес которых не превосходит  $t$  при некотором фиксированном  $t \leq [(d - 1)/2]$ . Для выполнения декодирования надо найти такое множество подстановок, чтобы код был инвариантен относительно этого множества и чтобы для каждого вектора  $\mathbf{e}$ , вес которого не превосходит  $t$ , нашлась бы подстановка, передвигающая все ошибки из информационных позиций в проверочные.

Метод перестановочного декодирования состоит в следующем. Определим оператор подстановок  $p_j$ . По полученному вектору  $\mathbf{y}$  вычисляются векторы сдвигов  $p_j \{ \mathbf{y} \}$  и их синдромы  $\mathbf{s}^{(j)}$  до тех пор, пока не будет найден индекс  $j$ , для которого вес  $wt(\mathbf{s}^{(j)}) \leq t$ . При этом все ошибки будут сосредоточены в первых  $r = n - k$  позициях вектора  $p_j \{ \mathbf{y} \}$  и задаются равенством  $[\mathbf{s}^{(j)}]^T = (e_0, \dots, e_{r-1})$ .

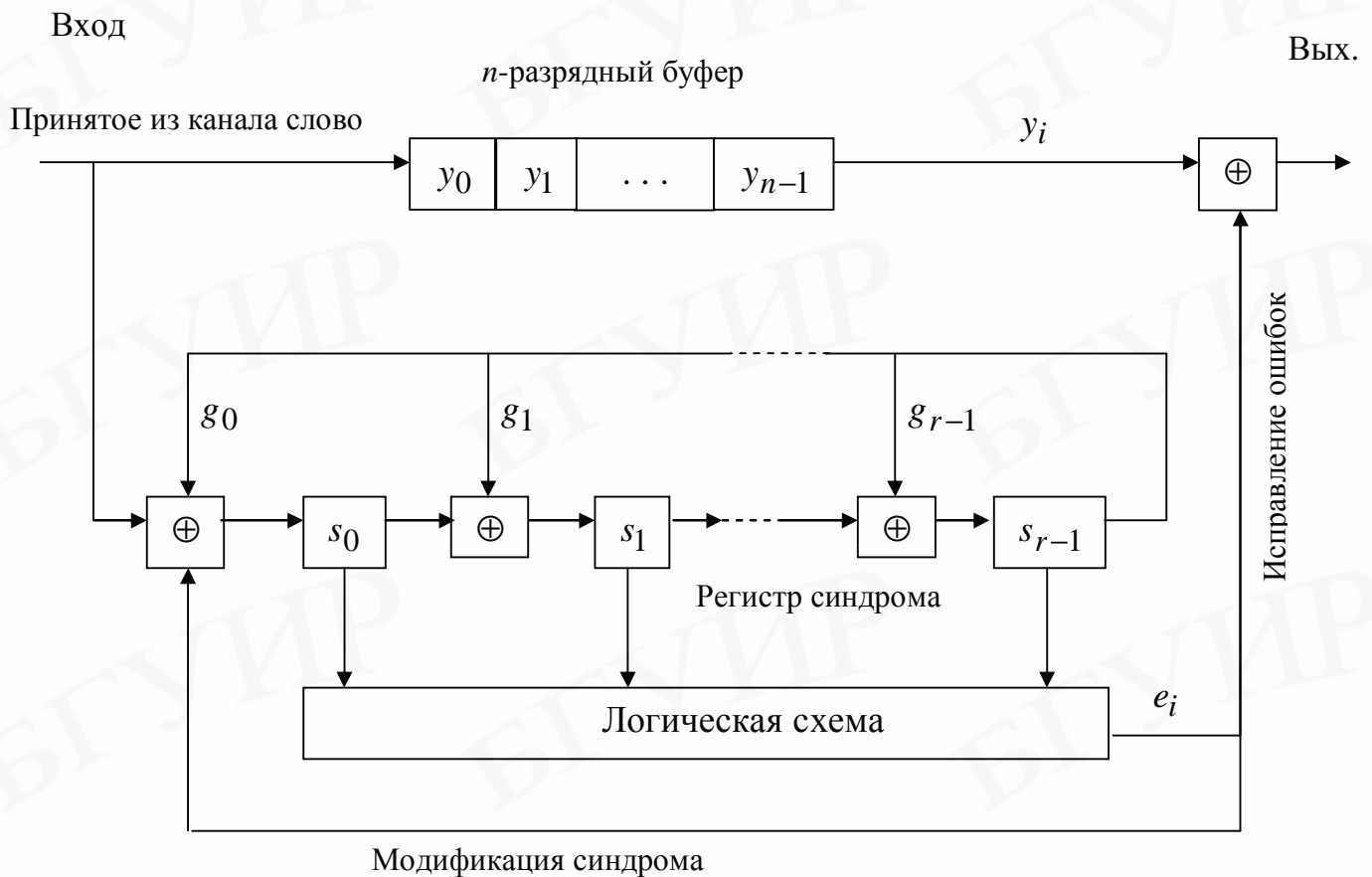


Рис. 5. Схема декодера Меггитта

Следовательно, принимаемый вектор декодируется как слово

$$\mathbf{c} = p_j^{-1} \{p_j \mathbf{y} + (e_0, \dots, e_{r-1}, 0, \dots, 0)\}.$$

Метод вылавливания ошибок является частным случаем перестановочного декодирования и использует циклические постановки  $D_j$ . Метод позволяет исправлять все векторы ошибок, содержащие круговую серию нулей длиной не менее  $k$ .

*Алгоритм*

1. Вычисляется синдром  $s(x)$  для принимаемого сигнала  $y(x)$ , используя алгоритм деления на порождающий полином.
2. Установка  $j := 0$ .
3. Если  $wt(s_j(x)) \leq t$ , тогда полагаем  $e(x) = x^j (s_j, \mathbf{0})$  и корректируем ошибку, вычисляя  $y(x) - e(x)$ .
4. Устанавливаем  $j := j + 1$ .
5. Если  $j = n$ , тогда алгоритм останавливается и ошибка считается не выловленной.
6. Если  $\deg(s_{j-1}(x)) < n - k - 1$ , тогда  $s_j(x) = x s_{j-1}(x)$ ; в противном случае –  $s_j(x) = x s_{j-1}(x) - g(x)$ .
7. Перейти к шагу 3.

*Пример.* Пусть  $g(x) = 1 + x^4 + x^6 + x^7 + x^8$  генерирует бинарный циклический код  $(15,7,5)$ , позволяющий исправлять две ошибки. Показать, что ошибка веса два, содержащая в своей структуре пачку из 7 нулей, может быть выловлена.

*Решение.* Предположим, что принимается вектор  $y = (1100\ 1110\ 1100\ 010)$ . Вычислим синдром  $y(x) = (x + x^2 + x^4 + x^5) g(x) + (1 + x^2 + x^5 + x^7)$ . Далее вычисляем синдромы  $s_i(x)$  для циклических сдвигов  $x^i y(x)$  до тех пор, пока вес синдрома не станет не более двух:  $wt(s_i(x)) \leq 2$ . Вычисления сведем в таблицу:

$i$	$s_i(x)$
0	10100101
1	11011001
2	11100111
3	11111000
4	01111100
5	00111111
6	00011111
7	10000100

Ошибка представляется как

$$e = x^{15-7}(s_7, \mathbf{0}) = x^8(10000100\ 00000000) = (0000\ 0000\ 1000\ 010).$$

Декодируем кодовое слово как

$$c = y - e = (1100\ 1110\ 0100\ 000).$$

*Декодирование пачки ошибок методом вылавливания.* Параметры корректирующего кода  $(n, k)$ , исправляющего пачки ошибок длиной  $t$ , должны удовлетворять условию  $(n - k) \geq 2t$ . Предполагается, что структура вектора пачки ошибок длиной  $t$  имеет отрезок из  $(n - t)$  нулевых элементов. Если вектор  $e$  представляет собой пачку ошибок длиной  $t$  и ошибки располагаются на первых  $(n - k)$  позициях вектора, тогда синдром  $H(e^T) = s$  характеризует структуру пачки ошибок длиной не более  $t$ . Если ошибки располагаются на позициях первых  $(n - k)$  символов вектора, то для вычисления оценки ошибки используется свойство циклического сдвига синдрома, как и в рассмотренном выше случае, только контролируется не вес, а используется его свойство (см. алгоритм I). Контролируется  $(n - k)$  первых позиций синдрома. Если конфигурация синдрома  $s_j(x)$  идентифицирует пачку ошибок длиной  $t$  или менее, то вектор ошибок  $e(x) = x^{n-j}(s_j, \mathbf{0})$ .

### 2.5.3. Мажоритарно декодируемые коды

Мажоритарно декодируемые коды образуют класс линейных кодов, имеющих ортогональные проверки и допускающих мажоритарное декодирование, основанное на принципе «голосование по большинству».

*Самоортогональные коды (СОК).* Такие коды характеризуются тем, что система всех проверок, контролирующих символ  $e_0$ , уже сама является ортогональной относительно данного символа. СОК задаются с помощью



порождающих полиномов  $g(x)$ , *разностные треугольники* (набор разностей между степенями с ненулевыми коэффициентами) которых не содержат одинаковых элементов.

*Пример.* Построить разностный треугольник для полинома  $g(x)=1+x+x^4+x^6$ .

*Решение.* Разностный треугольник имеет вид

$$\begin{array}{ccc} 1-0, 4-0, 6-0 & & 1, 4, 6 \\ 4-1, 6-1 & \Rightarrow & 3, 5 \\ 6-4 & & 2. \end{array}$$

Блочный СОК имеет кодовое расстояние  $d$ , равное увеличенному на 1 количеству ненулевых компонент порождающего полинома. Количество информационных символов  $k$  в блоковом СОК с кодовой скоростью  $R = 1/2$  равна как минимум  $(2m+1)$ , где  $m$  – максимальная степень порождающего полинома. Длина такого кода равна  $2k$ . В качестве примера можно привести код с параметрами  $n=26, k=13, R=0,5, d=5, g(x)=1+x+x^4+x^6$ .

Рассмотрим код БЧХ (15, 7), проверочная матрица которого получается путем транспонирования одного из цикла вычисления синдрома и имеет вид

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Используя 3, 7, (1+5) и (0+2+6) строки данной матрицы, можно составить проверочные уравнения, зависящие только от вектора ошибки  $e$ :

$$P_1 = e_3 + e_{11} + e_{12} + e_{14},$$

$$P_2 = e_7 + e_8 + e_{10} + e_{14},$$

$$P_3 = e_1 + e_5 + e_{13} + e_{14},$$

$$P_4 = e_0 + e_2 + e_6 + e_{14}.$$

Каждая компонента вектора ошибки входит только в одно уравнение, кроме  $e_{14}$ , входящего во все четыре уравнения. Проверочные уравнения, обладающие таким свойством, называются *ортгональными* по  $e_{14}$ . Одиночная ошибка, отличная от  $e_{14}$ , приведет к нарушению не более одного уравнения, в то время как ошибка  $e_{14}$  – к нарушению всех четырех уравнений. Аналогично любая комбинация двух ошибок, не содержащих  $e_{14}$ , приведет к нарушению не более двух уравнений, а содержащая  $e_{14}$  – не менее трех. Таким образом, используя только эти уравнения, можно исправить все двойные ошибки.

*Правило декодирования* состоит в вычислении всех четырех указанных проверок на четность и сравнения их суммы со значением 3. Если сумма больше или равна 3, то следует исправить ошибку в позиции 14. Для исправления всех остальных ошибок следует циклически сдвигать принятое кодовое слово и повторять процедуру.

Схема мажоритарного декодера для рассмотренного кода показана на рис. 6. Декодер работает аналогично декодеру Меггитта, вычисляя сперва синдром, а затем сдвигая его по крайней мере на один полный цикл для исправления ошибок. В схеме декодера может присутствовать обратная связь с выхода мажоритарного элемента. Если обратная связь не используется, то достаточно одного цикла – и последовательность может поступать к пользователю после исправления ошибок. Если обратная связь имеется, то для полной реализации возможностей декодирования должен быть выполнен по крайней мере еще один цикл. Наличие обратной связи позволяет исправлять дополнительные комбинации ошибок.

*Пример.* Синтезировать программу моделирования работы порогового декодера кода БЧХ (15, 7) с порождающим полиномом  $g(x) = 1 + x^4 + x^6 + x^7 + x^8$  (см. рис. 6). Информационное слово равно  $a(x) = x^6 + x^4 + x^3 + x^2 + 1$ . Полином ошибки равен  $e(x) = 1 + x^8$ .

*Решение.* Предварительно составим программу кодера систематического кода.

Определим информационный, порождающий полиномы и полином ошибки:

$$a := x^6 + x^4 + x^3 + x^2 + 1, g := 1 + x^4 + x^6 + x^7 + x^8, e := 1 + x^8.$$

Сформируем полином кодового слова:

$$> c := \text{sort}(\text{expand}(a * x^8 + \text{rem}(a * x^8, g, x)) \bmod 2);$$

$$c := x^{14} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^6 + x^5 + x^4 + x^3.$$

Внесем в структуру кода ошибки, вычислив сумму  $y(x) = c(x) + e(x) \bmod 2$ :

$$> y := c + e \bmod 2;$$

$$y := x^{14} + x^{12} + x^{11} + x^{10} + x^7 + x^6 + x^5 + x^4 + x^3 + 1.$$

Преобразуем полученный полином в вектор:

$$> yv := \text{ToVector}(y, x, 15, 2);$$

$$yv := [1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 0, 1].$$

Составим программу вычисления синдрома с помощью регистра сдвига с обратными связями.

Вычислитель синдрома –  
регистр сдвига с обратными связями

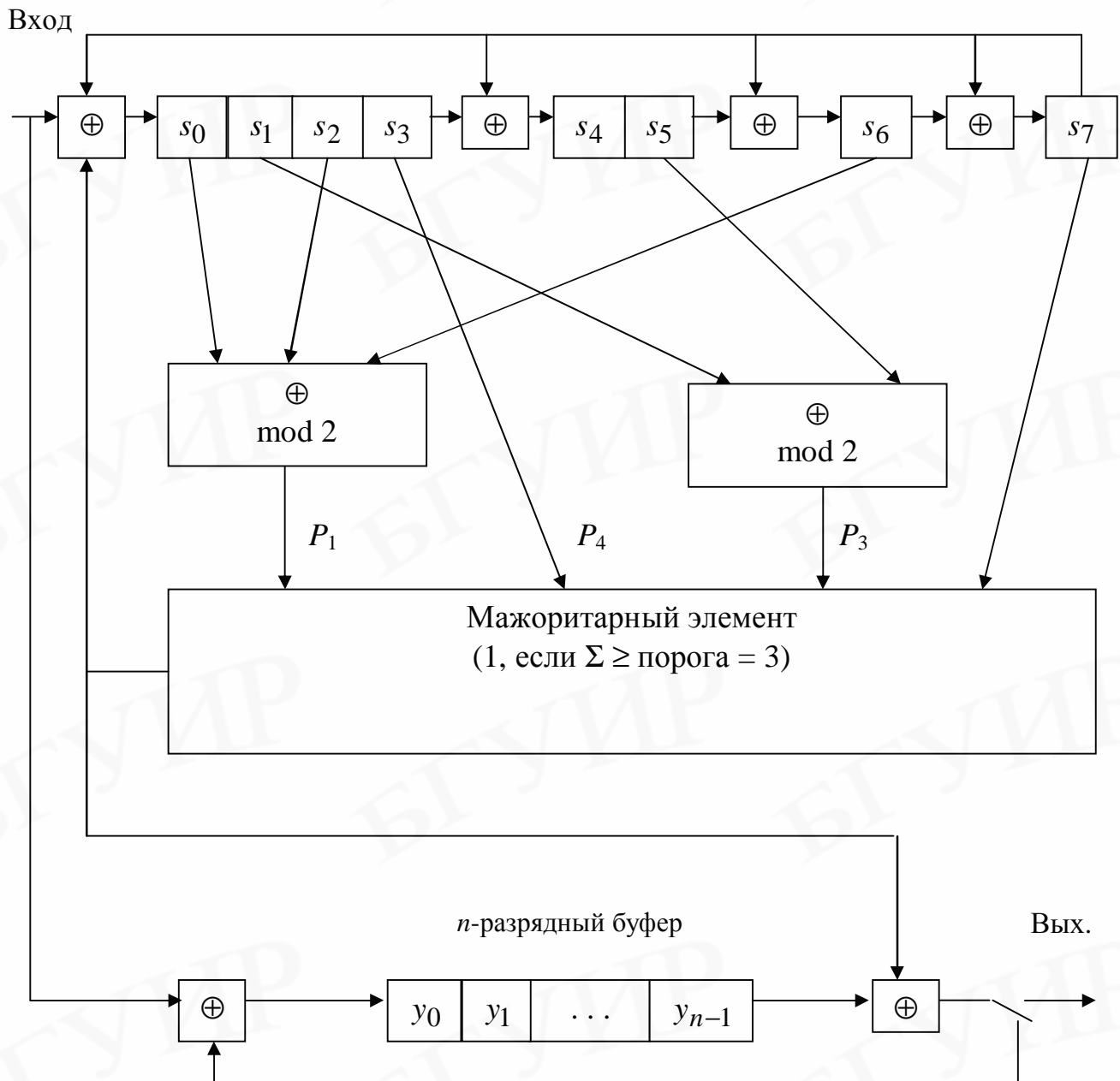


Рис. 6. Мажоритарный декодер кода

Зададим функцию обратной связи:

```
> g0 := 1; g1 := 0; g2 := 0; g3 := 0; g4 := 1; g5 := 0; g6 := 1; g7 := 1; g8 := 1;  
> fm := (c8, c7, c6, c5, c4, c3, c2, c1) -> (c1 + bm mod 2, (c8 + g1*c1 mod 2), (c7 +  
+ g2*c1 mod 2), (c6 + g3*c1 mod 2), (c5 + g4*c1 mod 2), (c4 + g5*c1 mod 2), (c3 +  
+ g6*c1 mod 2), (c2 + g7*c1 mod 2));  
> c := (0, 0, 0, 0, 0, 0, 0, 0);
```

Составим программу и проведем вычисления синдрома:

```
> for j from 1 to 15 do  
  bm := yv[15 - j + 1]; c := fm(c); s := c; if j = 15 then print ('Синдром = ', s); fi;  
od;
```

Вычисления дают результат:

*Синдром*  $s = 0, 0, 0, 0, 1, 0, 1, 1$ .

Напишем программу циклических сдвигов синдрома регистром сдвига с обратными связями и работы мажоритарного элемента:

```
> for j from 1 to 15 do  
  mj := op(4, [s]) + (op(2, [s]) + op(6, [s]) mod 2) + (op(1, [s]) + op(3, [s]) +  
  + op(4, [s]) mod 2) + op(8, [s]); s := fd(s);  
  if mj >= 3 then print ('Вых Maj = ', mj, 'Позиция ошибки = ', 15 - j); fi; od;
```

Проведя вычисления, получим координаты ошибок:

*Вых. Maj* = 3, *Позиция ошибки* = 8,

*Вых. Maj* = 3, *Позиция ошибки* = 0.

Сравнивая полученный результат с заданной конфигурацией ошибок, можно сделать вывод, что декодер верно оценил позиции ошибок.

## 2.6. Некоторые циклические коды с особыми свойствами

*Коды максимальной длины.* Коды максимальной длины дуальны кодам Хэмминга длиной  $n = 2^m - 1$ , порождающим многочленом которого является примитивный полином  $g(x) = p(x)$ . Дуальный код такой же длины получается, если полином  $p(x)$  взять в качестве проверочного  $h(x) = p(x)$ . Этот многочлен может быть использован для построения цепей обратных связей в кодере на регистре сдвига с  $k$  ячейками (см. рис. 3).

Для кода максимальной длины все  $2^m - 1$  ненулевых кодовых слов являются циклическими сдвигами одного ненулевого кодового слова, так что все кодовые слова имеют один и тот же вес. Такой код называется *эквиликулированным* или *симплексным*. Кодер длинного кода производит последовательности с хорошими свойствами случайности (псевдослучайные  $m$ -последовательности).

*Квадратично-вычетные коды.* Пусть  $p$  – нечетное простое число. Классы вычетов чисел  $1^2, 2^2, 3^2, \dots$  по модулю  $p$  называются квадратичными вычетами по модулю  $p$ . Так, при  $p = 17$  квадратичными вычетами будут числа 1, 4, 9, 16, 8, 2, 15, 13. Остальные числа 3, 5, 6, 7, 10, 11, 12, 14 будут невычетами.

Квадратично-вычетные (КВ) коды определяются требованием, согласно которому корнями порождающего многочлена должны быть в точности все квадратичные вычеты.

Пусть  $R$  – множество квадратичных вычетов по модулю числа  $n = 8m \pm 1$  и  $N$  – множество невычетов. Зададим порождающие многочлены:

$$g_r(x) = \prod_{i \in R} (x - a^i), \quad g_n(x) = \prod_{i \in N} (x - a^i),$$

где  $\alpha$  – элемент порядка  $n$  в расширении поля  $GF(2)$ .

Коды, порожденные многочленами  $g_r(x)$ ,  $(x - 1) g_r(x)$ ,  $g_n(x)$ ,  $(x - 1) g_n(x)$ , называются квадратично-вычетными. При  $n = 8m \pm 1$  выполнено равенство

$$(x^n - 1) = (x - 1) g_r(x) g_n(x).$$

Кодовое расстояние КВ-кодов удовлетворяет неравенству  $d^2 > n$ .

*Пример.* Построить КВ-код  $n = 17$ .

*Решение.* Рассмотрим поле  $GF(2^8)$ . Число  $255 = 15 \cdot 17$ . Пусть  $\beta$  – примитивный элемент поля. Тогда  $\beta^{15}$  имеет порядок 17. В качестве корней  $g_r(x)$  можно взять элементы поля

$$a, a^2, a^4, a^8, a^{16}, a^{32} = a^{15}, a^{30} = a^{13}, a^{26} = a^9, a^{18} = a.$$

Построим поле  $GF(2^8)$  с помощью неприводимого полинома  $x^8 + x^4 + x^3 + x^2 + 1$ . Прделав необходимые вычисления, получим

$$g_r(x) = x^8 + x^7 + x^6 + x^4 + x^2 + x + 1.$$

Заметим, что  $g_r(x)$  является минимальным полиномом для  $\alpha = \beta^{15}$ .

Другим примером КВ кода является двоичный код Голя, который обнаружил, что сумма

$$\sum_{i=0}^3 C_{23}^i = 2^{11}$$

удовлетворяет границе совершенных кодов. Равенство позволяет предположить возможность существования совершенного двоичного (23, 12, 7) кода с  $t = 3$  – возможностью исправлять до трех ошибок в словах длиной 23 символа. Такой код существует. Порождающие многочлены задаются формулами:

$$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1 \text{ и } g(x) = x^{11} + x^{10} + x^6 + x^5 + x^4 + x^2 + 1.$$

*CRC-коды.* Примером использования семейства циклических кодов является контроль ошибок. При передаче данных в пакетных режимах эти коды используются для контроля целостности в блоках данных. Пусть  $p(x)$  – примитивный многочлен степени  $m$ , тогда порождающий полином CRC-кода  $g(x)$  можно записать в виде произведения

$$g(x) = (1 + x)p(x).$$

С помощью порождающего полинома  $g(x)$  может быть построен циклический CRC-код с параметрами  $n = 2^m - 1$ ,  $k = 2^m - m - 2$ , имеющий  $m + 1$  проверочных символов и  $d_{\min} = 4$ .

### 3. МОДЕЛИРОВАНИЕ СИСТЕМ ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ

#### 3.1. Моделирование кодовых структур в среде Maple

Моделирование выполняется с помощью собственных процедур Maple, а также из библиотек 'codes.mpl' и «Циклические коды».

*Собственные процедуры* позволяют подключать программные процедуры линейной алгебры «with(linalg)», полиномиальных вычислений «with(polytools)», теории чисел «with(numtheory)».

*Библиотека 'codes.mpl'* содержит следующие процедуры, удобные для формирования и исследования кодовых структур:

1. `iscyclic(pol, n, p)`, где `pol` – заданный полином; `n` – длина кода; `p` – основание системы (количество элементов поля). Осуществляет проверку полинома на возможность использования его для генерации циклического кода.

2. `poly2coeff_list(pol)`, где `pol` – задающий полином. Выполняет выделение коэффициентов из полинома:

3. `coeff_list2poly(m)`, где `m` – список коэффициентов. Создает полином по списку коэффициентов.

4. `generator_matrix(pol, n, p)`, где `pol` – задающий полином; `n` – длина кода; `p` – основание системы (количество элементов поля). Формирует порождающую матрицу циклического кода.

5. `check_matrix(pol, n, p)`, где `pol` – задающий полином; `n` – длина кода; `p` – основание системы (количество элементов поля). Формирует проверочную матрицу циклического кода.

6. `code_list(G, p)`, где `G` – порождающая (генераторная) матрица; `p` – число элементов поля. Находит все кодовые слова по порождающей матрице.

7. `sort(C, weight_order)` – сортировка кодовых слов в соответствии с их весом.

8. `weight_enumerator_vector(G, p)`, где `G` – порождающая (генераторная) матрица; `p` – число элементов поля. Находит количество кодовых слов с одинаковым весом.

9. `min_distance(G, p)`, где `G` – порождающая (генераторная) матрица; `p` – число элементов поля. Находит минимальное кодовое расстояние.

10. `matrix_times_vector modp(H, v2, 2)`, где `G` – матрица; `v` – вектор; `p` – число элементов поля. Умножает матрицу на вектор.

11. `decode(v, G, p, d)`, где `v` – декодируемое слово; `G` – порождающая матрица; `p` – количество элементов поля; `d` – кодовое расстояние. Декодирует кодовое слово для любого кода Хэмминга для поля  $GF(p)$ .

*Библиотека «Циклические коды»* содержит следующие процедуры:

1. `ToPoly(w, p, var)`, где `w` – упорядоченное множество (array), `p` – основание системы счисления, `var` – символ формальной переменной. Преобразует вектор `w` в полином.

2. ToVector(poly, var, n, p), где poly – полином, var – символ формальной переменной, n – требуемый размер вектора, p – основание системы счисления. Преобразует полином в векторную форму.

### 3.2. Моделирование в среде Electronics Workbench

Экспериментальная часть лабораторной работы выполняется на ПЭВМ в среде схемотехнического моделирования Electronics Workbench.

Модель лабораторной установки находится в файле **cyclic\_15\_11.ewb** и содержит источник двоичного сообщения, кодеры разделимого и неразделимого кодов, модель канала связи, синдромный декодер, логический анализатор для отображения состояний в различных точках схемы (рис. 7). Включение питания производится тумблером в верхнем правом углу окна программы. Переключение кодеров осуществляется нажатием клавиши «S». Данная схема позволяет моделировать процесс передачи информации в системе с помехоустойчивым кодированием при возникновении заданной ошибки в канале связи. Используется код (15, 11) с порождающим полиномом  $g(x) = x^4 + x + 1$ . Путем внесения незначительных изменений в схему может быть реализован другой циклический код длиной 15. Для этого необходимые модули открываются двойным нажатием левой кнопки мыши, после чего возможно графическое редактирование схемы.

Передаваемое двоичное сообщение задается в 16-разрядном генераторе слов (рис. 8). Для этого необходимо настроить генератор, как показано на рисунке, выделить первое слово в списке слева, затем внизу в строке **Binary** ввести 11 информационных разрядов, дополнив их справа пятью нулями. Старший разряд MSB находится слева, младший LSB – справа. Модуль **p\_to\_s** преобразует информационное слово из параллельного кода в последовательный, выдаваемый начиная со старших разрядов.

Кодер неразделимого кода (рис. 9) собран в модуле **coder1** и представляет собой 4-разрядный регистр сдвига (на D-триггерах) с внешними сумматорами. Регистр тактируется внешним сигналом CLK. Перед началом кодирования каждого слова осуществляется обнуление регистра по внешнему сигналу сброса RESET~.

Кодер разделимого кода (рис. 10) собран в модуле **coder2** и представляет собой 4-разрядный регистр сдвига (на D-триггерах) с обратной связью. Регистр тактируется внешним сигналом CLK. Перед началом кодирования каждого слова осуществляется обнуление регистра по внешнему сигналу сброса Reset. Схема содержит также два ключа, собранных на элементах И, ИЛИ, НЕ, управляемых внешним сигналом Timer 2.

Модель канала связи (рис. 11) собрана в модуле **Channel** и содержит сумматор, осуществляющий последовательное поразрядное сложение кодового слова и вектора ошибки. Последний задается путем подачи логических единиц и нулей на входы блока **p\_to\_s**, для чего эти входы подключаются к источнику

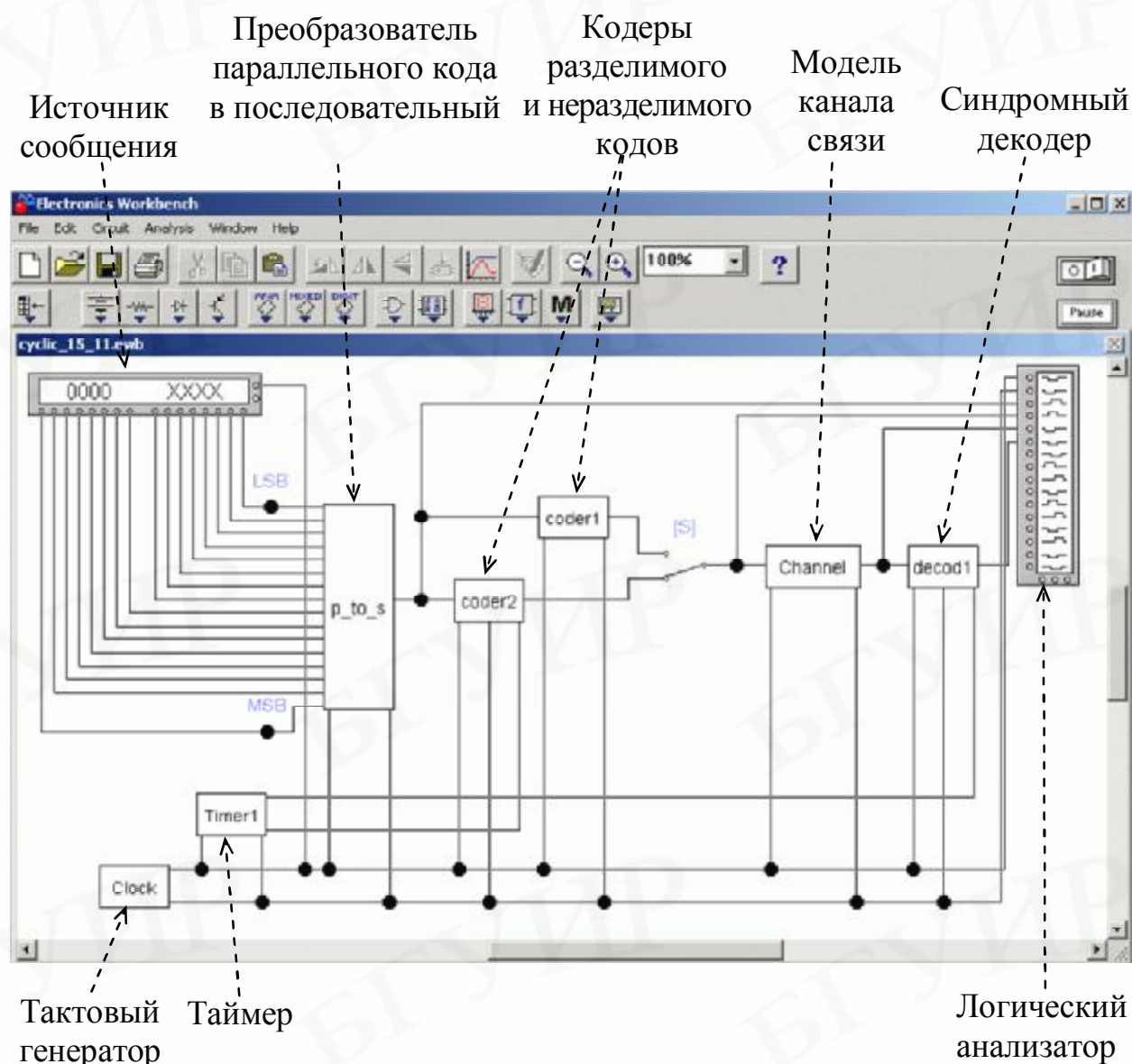


Рис. 7. Модель лабораторной установки в среде Electronics Workbench

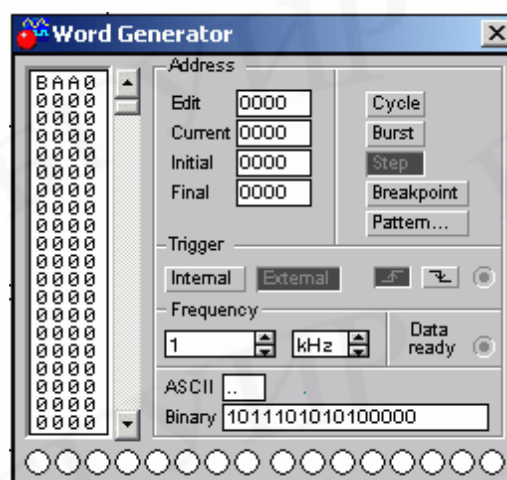


Рис. 8. Задание информационного сообщения в генераторе слов



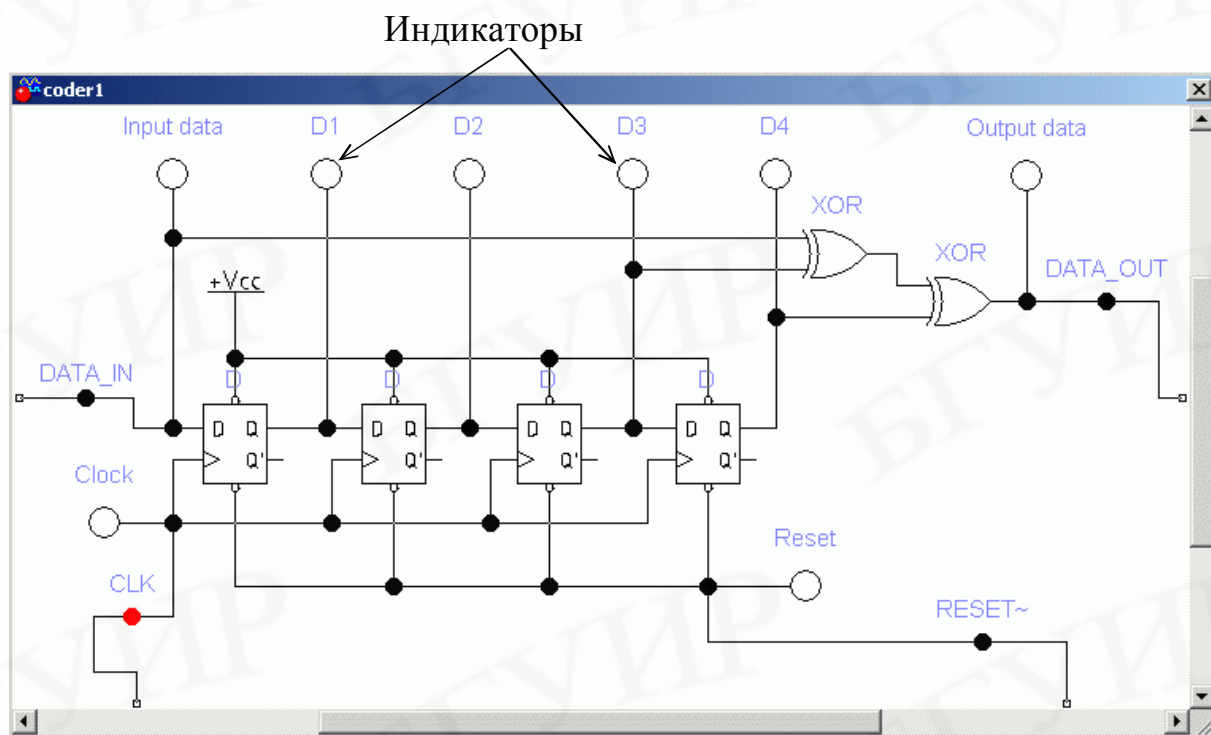


Рис. 9. Кодер неразделимого кода

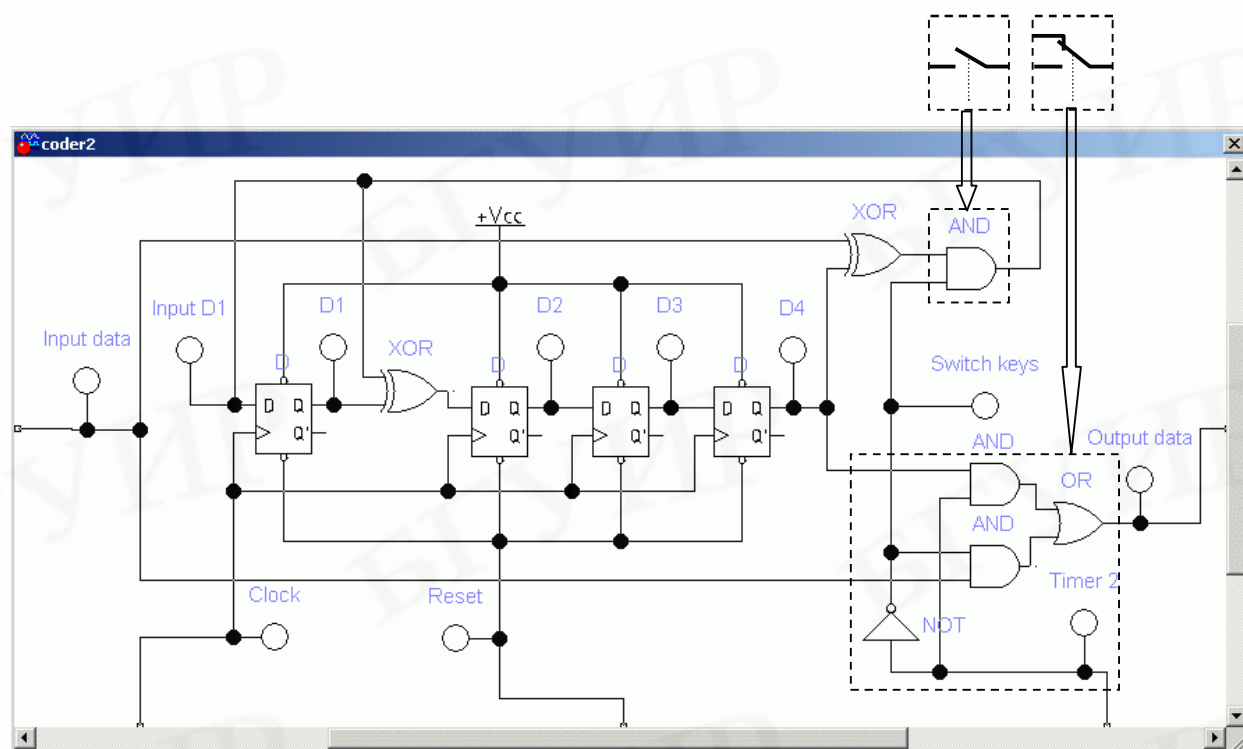


Рис. 10. Кодер разделимого кода

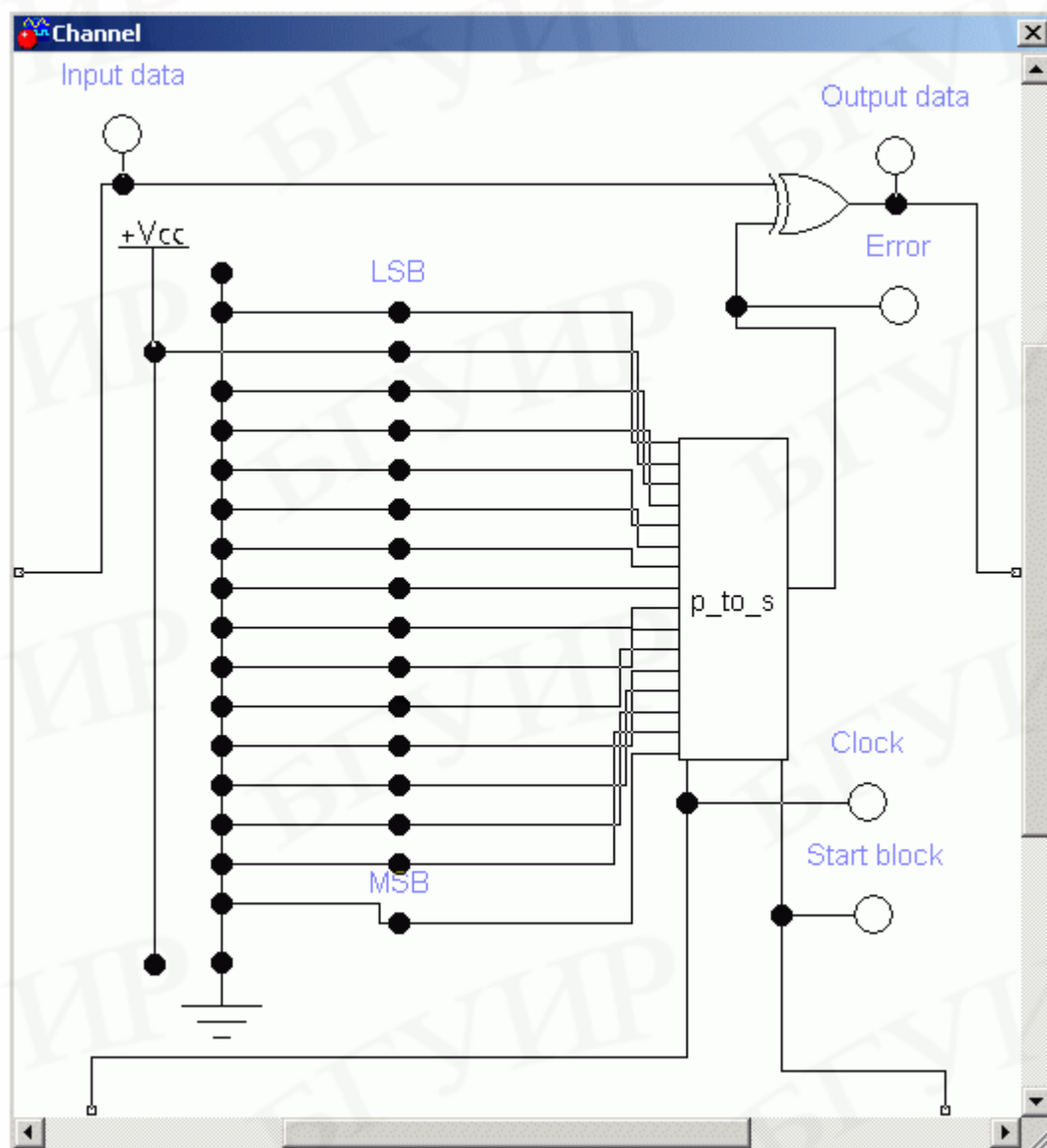


Рис. 11. Задание вектора ошибки в канале связи

питания или к общей шине. Поскольку вектор ошибки имеет длину 15, то самый верхний по схеме вход блока **p\_to\_s** LSB не используется и должен быть подключен к общей шине.

Синдромный декодер (рис. 12) собран в модуле **decod1** и содержит генератор синдрома (регистр с обратной связью), дешифратор (селектор кода 0001), ключ, управляемый по внешнему сигналу Switch key, 15-разрядный регистр сдвига и корректирующий сумматор.

Опорный генератор собран в модуле **Clock**. Он вырабатывает тактовый сигнал CLK, а также инверсный сигнал сброса Reset, формируемый на каждом 31-м такте перед началом передачи очередного слова по каналу связи.

Для управления ключами кодера и декодера используются два таймера, собранные в модуле **Timer1** (рис. 13). Они построены на базе 4-разрядного двоичного счетчика, к выходам QA, QB, QC и QD которого подключены селекторы кода.

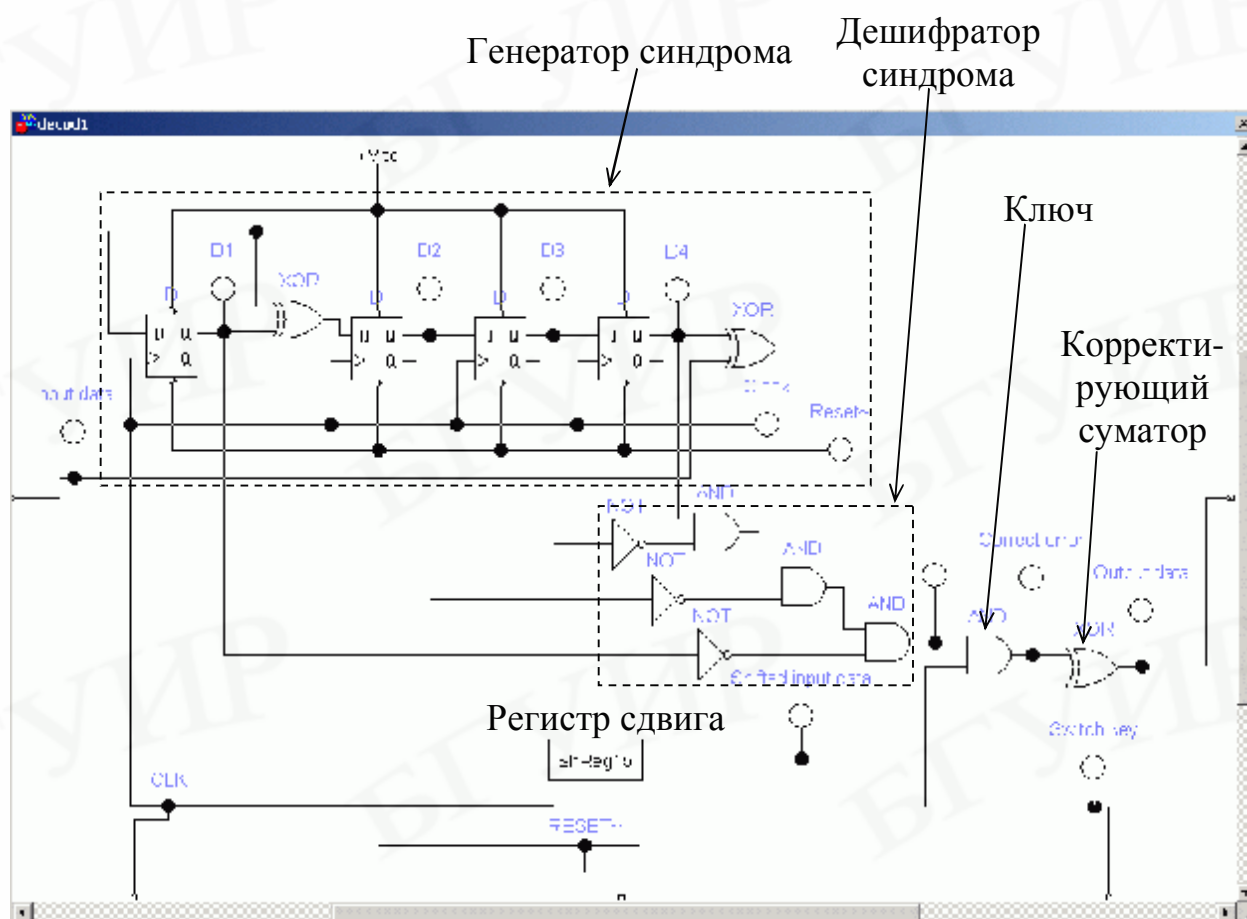


Рис. 12. Синдромный декодер

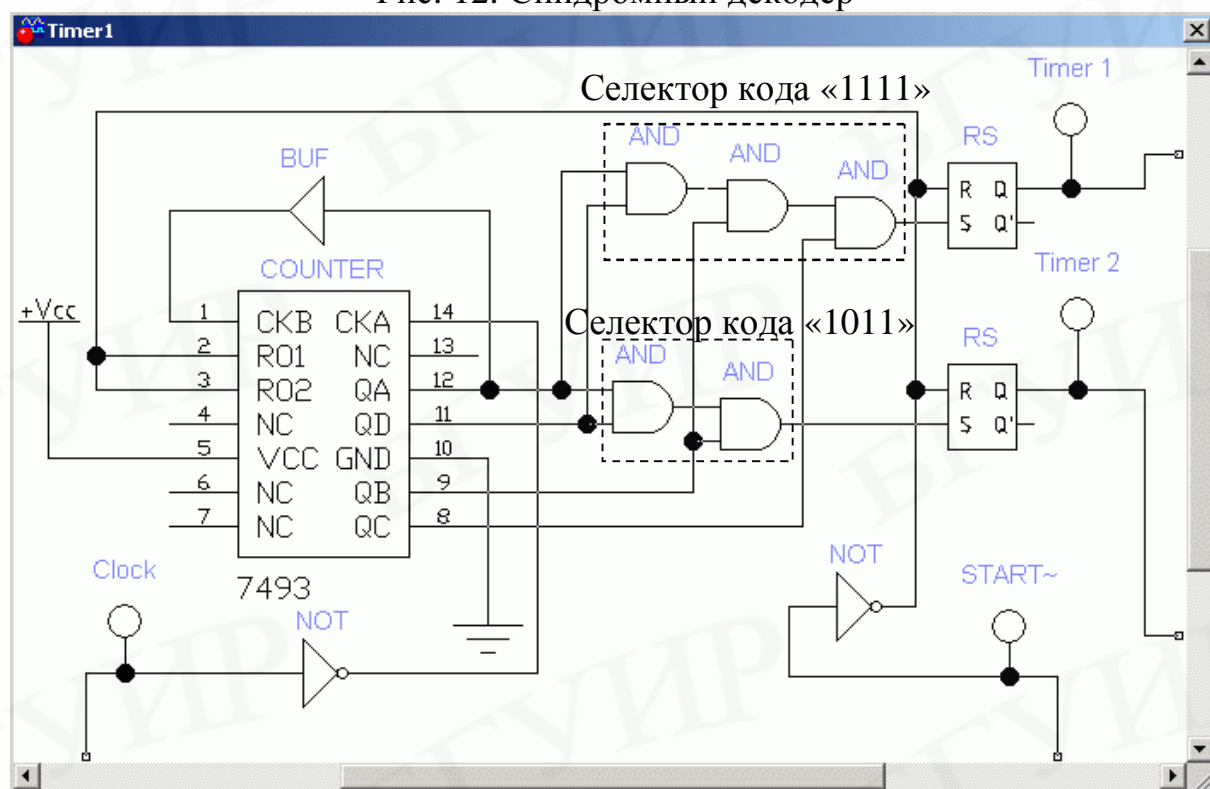


Рис. 13. Таймеры

Выходные сигналы Timer 1 и Timer 2 формируются RS-триггерами, сброс которых происходит по сигналу  $START\sim$  перед началом передачи очередного слова по каналу связи. Момент установки каждого триггера определяется настройкой соответствующего селектора кода. Так верхний триггер на рис. 14 установится в 1 на 15-м такте, а нижний – на 11-м. Если количество информационных разрядов кода не равно 11, то необходимо изменить настройку нижнего селектора.

После включения лабораторной установки логический анализатор (рис. 14) выводит осциллограммы тактового сигнала (CLK), сигнала сброса (RESET $\sim$ ), информационного сообщения ( $a$ ), закодированного сообщения ( $c$ ), принятого из канала кодового сообщения ( $y$ ), декодированного сообщения ( $\hat{c}$ ).

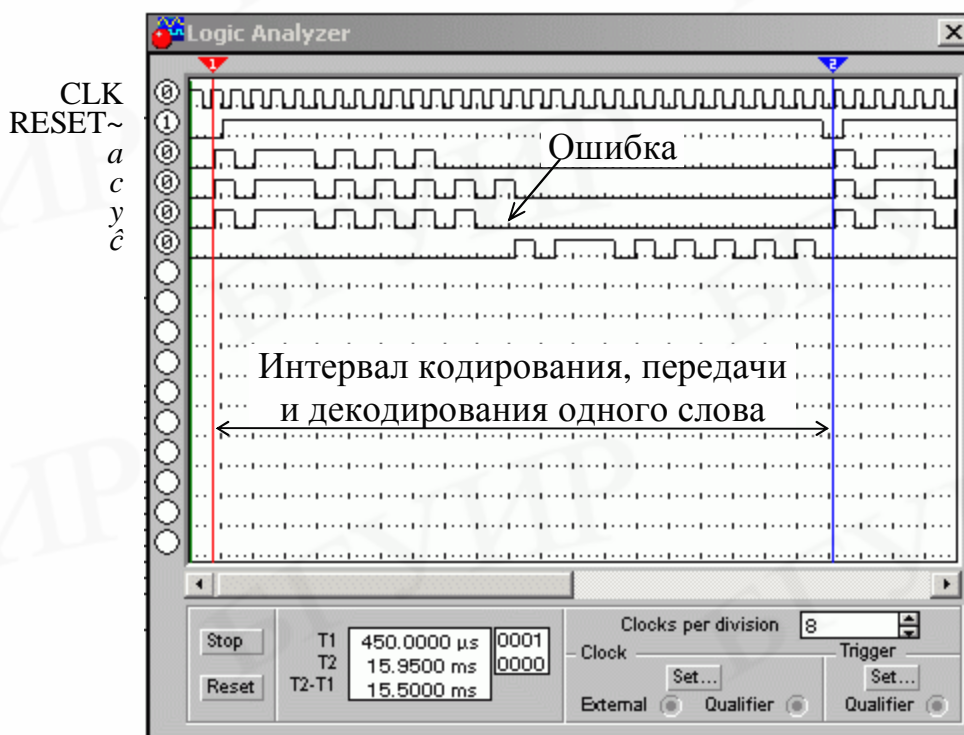


Рис. 14. Логический анализатор

## 4. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ

### 4.1. Предварительное задание

1. Получить у преподавателя исходные данные: порождающий полином  $g(x)$ , длину кода  $n$ , информационное сообщение  $a$ .
2. Определить количество информационных ( $k$ ) и проверочных ( $r$ ) символов кода.
3. Определить количество кодовых слов кода ( $M$ ).
4. Закодировать сообщение  $a$  разделимым и неразделимым кодом.
5. Вычислить проверочный полином  $h(x)$ .
6. Построить порождающую ( $G$ ) и проверочную ( $H$ ) матрицы.

## 4.2. Лабораторное задание

*Исследование передачи информации в системе с помехоустойчивым кодированием*

1. Запустить программу **Electronics Workbench v.5.0** и открыть файл со схемой лабораторной установки.
2. В генераторе слов задать информационное сообщение  $a$ , а в модуле **Channel** – вектор одиночной ошибки в любом разряде, соблюдая инструкции разд. 3 пособия.
3. Установить ключ «S» в верхнее положение, чтобы в канал подавался неразделимый код. Включить моделирование схемы, открыть логический анализатор и зарисовать осциллограммы на интервале передачи одного слова. На осциллограммах найти символы информационного и кодового слова, позицию ошибки, убедиться, что ошибка исправлена декодером. Выключить схему.
4. Повторить п. 3, установив ключ «S» в нижнее положение, чтобы в канал подавался разделимый код.
5. Повторить п. 4, предварительно задав в модуле **Channel** вектор двойной ошибки.

*Изучение процесса кодирования*

6. Открыть модуль **coder1** (кодер неразделимого кода). Включить моделирование схемы и снять (записать) таблицу состояний на 15 тактах работы кодера в контрольных точках «Input data», «D1», «D2», «D3», «D4», «Output data». Для этого останавливать моделирование после каждого такта нажатием кнопки «Pause» на панели инструментов. Такты контролировать по индикатору «Clock». Кодирование начинается после установления сигнала RESET~ в «1».
7. Аналогичным образом снять таблицу состояний кодера разделимого кода (модуль **coder2**) на 15 тактах его работы. Контрольные точки: «Input data», «Input D1», «D1», «D2», «D3», «D4», «Output data», «Switch key».

*Изучение процесса декодирования*

8. В модуле **Channel** задать одиночную ошибку. Подключить к каналу связи выход кодера разделимого кода. Открыть модуль синдромного декодера (**decod1**). Включить моделирование схемы и снять таблицу состояний на 30 тактах работы декодера в контрольных точках «Input data», «D1», «D2», «D3», «D4», «Correct error», «Shifted input data», «Output data», «Switch key».

## 4.3. Расчетная часть

Задан циклический код с параметрами:

Вариант 1:  $n = 26$ ,  $R = 0,5$  и  $g(x) = x^6 + x^4 + x + 1$ .

Вариант 2: задается преподавателем.

Требуется:

1. Построить схему (алгоритм) кодера систематического кода.
2. Сформировать кодовое слово по заданной информационной части.
3. Построить порождающую и проверочную матрицы.
4. Построить схему (алгоритм) декодера по заданному методу декодирования (для варианта 1 – мажоритарный декодер).
5. Проверить работу декодера для случая, когда принятое слово искажено вектором ошибки (вектор ошибки задается при получении задания).
6. Найти максимальное количество исправляемых (обнаруживаемых) ошибок.

## 5. СОДЕРЖАНИЕ ОТЧЕТА

1. Цель работы, формулировка исследовательских задач.
2. Расчеты, программы вычислений по предварительному заданию.
3. Осциллограммы и таблицы состояний кодеров и декодера.
4. Расчетная часть.
5. Выводы.

## 6. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Поясните работу схем кодирования циклических кодов (см. рис. 1–3).
2. Напишите программы кодирования информации систематическим и несистематическим кодами.
3. Найдите порождающие многочлены циклических кодов длинами 15 и 31, исправляющих три ошибки на основе факторизации бинома  $x^n - 1$ .
4. Сформулируйте основные свойства функции синдрома циклического кода.
5. Составьте алгоритм и напишите программу работы декодера Меггитта для кода Хэмминга (15, 11).
6. Постройте и поясните работу схемы кодирования и вычисления синдрома CRC кода (7, 3).
7. Постройте и поясните работу декодирования кода Голея по методу выявления ошибок.
8. Постройте возможные алгоритмы декодирования пакета ошибок.
9. Циклический код (21, 11), порождаемый многочленом  $g(x) = x^{10} + x^7 + x^6 + x^4 + x^2 + 1$ , декодируется с помощью мажоритарного алгоритма. Найдите проверочные соотношения, ортогональные по первому ошибочному символу, и построьте схему декодера.
10. Рассмотрите (15, 4) код максимальной длины. Каково его кодовое расстояние? Нарисуйте схемы кодера с регистром сдвига и варианты схем декодеров различного типа.
11. Объясните работу кодера делимого кода по тактам.
12. Объясните работу синдромного декодера по тактам.

## ЛИТЕРАТУРА

1. Мак-Вильямс, Ф. Дж. Теория кодов, исправляющих ошибки / Ф. Дж. Мак-Вильямс, Н. Дж. Слоэн. – М. : Связь, 1979.
2. Кларк, Дж. Кодирование с исправлением ошибок в системах цифровой связи / Дж. Кларк, Дж. Кейн. – М. : Радио и связь, 1987.
3. Морелос-Сарагоса, Р. Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение / Р. Морелос-Сарагоса. – М. : Техносфера, 2005.
4. Золотарев, В. В. Помехоустойчивое кодирование. Методы и алгоритмы : справочник / В. В. Золотарев, Г. В. Овечкин. – М. : Горячая линия – Телеком, 2004.
5. Саломатин, С. Б. Кодирование в радиоэлектронных системах : учеб. пособие. – Минск : БГУИР, 2003.
6. Бурак, А. И. Моделирование импульсных и цифровых устройств в среде Electronics Workbench : метод. пособие / А. И. Бурак. – Минск : БГУИР, 2000.

## СОДЕРЖАНИЕ

<b>1. ЦЕЛЬ ЛАБОРАТОРНОЙ РАБОТЫ.....</b>	<b>3</b>
<b>2. КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ.....</b>	<b>3</b>
2.1. Циклические коды.....	3
2.2. Кодирование циклических кодов.....	7
2.3. Алгоритмы вычисления остатка от деления двух полиномов.....	11
2.4. Синдромные полиномы .....	12
2.4.1. Пакеты ошибок .....	13
2.5. Методы декодирования циклических кодов.....	13
2.5.1. Декодеры Меггитта.....	13
2.5.2. Перестановочное декодирование и метод вылавливания ошибок.....	14
2.5.3. Мажоритарно декодируемые коды .....	16
2.6. Некоторые циклические коды с особыми свойствами .....	20
<b>3. МОДЕЛИРОВАНИЕ СИСТЕМ ПОМЕХОУСТОЙЧИВОГО КОДИРОВАНИЯ .....</b>	<b>22</b>
3.1. Моделирование кодовых структур в среде Maple.....	22
3.2. Моделирование в среде Electronics Workbench.....	23
<b>4. ПОРЯДОК ВЫПОЛНЕНИЯ ЛАБОРАТОРНОЙ РАБОТЫ .....</b>	<b>28</b>
4.1. Предварительное задание.....	28
4.2. Лабораторное задание .....	29
4.3. Расчетная часть .....	29
<b>5. СОДЕРЖАНИЕ ОТЧЕТА.....</b>	<b>30</b>
<b>6. КОНТРОЛЬНЫЕ ВОПРОСЫ .....</b>	<b>30</b>
<b>ЛИТЕРАТУРА.....</b>	<b>31</b>



Учебное издание

**Саломатин Сергей Борисович**  
**Семашко Павел Геннадьевич**

## **ИССЛЕДОВАНИЕ ЦИКЛИЧЕСКИХ КОДОВ**

Методическое пособие  
по курсам

«Теория кодирования и защита информации»,  
«Цифровая обработка сигналов и прикладная теория кодирования»  
для студентов радиотехнических специальностей  
всех форм обучения

Редактор Н. В. Гриневич  
Корректор М. В. Тезина

---

Подписано в печать 08.04.2008.  
Гарнитура «Таймс».  
Уч.-изд. л. 1,5.

Формат 60×84 1/16.  
Печать ризографическая.  
Тираж 120 экз.

Бумага офсетная.  
Усл. печ. л. 2,09.  
Заказ 18.

---

Издатель и полиграфическое исполнение: Учреждение образования  
«Белорусский государственный университет информатики и радиоэлектроники»  
ЛИ №02330/0056964 от 01.04.2004. ЛП №02330/0131666 от 30.04.2004.  
220013, Минск, П. Бровки, 6