Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего профессионального образования
«Владимирский государственный университет
имени Александра Григорьевича и Николая Григорьевича Столетовых»

В. Г. ЖУРАВЛЕВ
Н. Ю. КУРАНОВА
Ю. Ю. ЕВСЕЕВА

# ПОМЕХОУСТОЙЧИВЫЕ КОДЫ

Учебное пособие


Владимир 2013

УДК 681.142.2
ББК 32.811.4
П56

Рецензенты:
Кандидат физико-математических наук, доцент кафедры информатики Владимирского государственного университета им. А. Г. и Н. Г. Столетовых
*А. А. Жукова*

Кандидат физико-математических наук, доцент
начальник отдела высшего профессионального образования Владимирского филиала Российской академии народного хозяйства и государственной службы при Президенте Российской Федерации
*И. В. Сидорова*

Печатается по решению редакционно-издательского совета ВлГУ

**Помехоустойчивые** коды : учеб. пособие / В. Г. Журавлев, Н. Ю. Куранова, Ю. Ю. Евсеева ; Владим. гос. ун-т им. А. Г. и Н. Г. Столетовых. – Владимир : Изд-во ВлГУ, 2013. – 96 с.
П56
ISBN 978-5-9984-0387-3

В пособии дается классификация разновидностей, характеристик и параметров корректирующих кодов, изложены принципы построения, формирования и обработки помехоустойчивых циклических кодов, их определение, свойства и эффективность, а также реализация кодирующих и декодирующих устройств этих кодов. Приводятся примеры построения циклических кодов.

Предназначено для студентов бакалавриата по направлению 050100 – Педагогическое образование, профиль «Математика и информатика». Разработаны на основе рабочей программы по дисциплине «Прикладная математика», составленной в соответствии с ФГОС ВПО третьего поколения. Могут быть использованы на лекциях, практических занятиях и для индивидуальной работы студентов в процессе самоподготовки.

Рекомендовано для формирования профессиональных компетенций в соответствии с ФГОС 3-го поколения.

Ил. 10. Табл. 5. Библиогр.: 18 назв.

УДК 681.142.2
ББК 32.811.4

ISBN 978-5-9984-0387-3

# Введение

Цель учебного пособия – помощь при изучении студентами положений, устанавливающих потенциальные возможности различных методов передачи, обработки и хранения информации. Важнейшая задача теории информации – разработка принципов построения оптимальных систем передачи информации со скоростью, близкой к пропускной способности канала, и со сколь угодно высокой достоверностью.

Задачами данного пособия является приобретение студентами:

– знаний в области конструирования оптимальных кодов с максимально возможным числом используемых сложных сигналов (кодовых комбинаций) при максимально возможном минимальном кодовом расстоянии;

– умения применять на практике фундаментальные положения основных теорем теории информации;

– умения выбирать энергоэффективные помехоустойчивые коды, грамотно выполнять расчет их помехоустойчивости в заданном канале связи, реализовывать их алгоритмы кодирования и декодирования.

Материал, изложенный в пособии, изучается в курсе дисциплины «Прикладная математика», которая относится к вариативной части профессионального цикла. Изучение проводится на 5-м курсе бакалавриата, включая 34 часа лекций и 38 часов практических занятий, базируется на основах ранее изученных дисциплин: «Математический анализ», «Линейная алгебра и аналитическая геометрия», «Дискретная математика». Знания и умения, полученные в результате изучения данного курса, могут быть использованы в дипломном проектировании, а также в дисциплинах, связанных с информатикой.

Процесс изучения темы „Помехоустойчивые коды” направлен на формирование следующих компетенций:

– владения культурой мышления, способности к обобщению, анализу, восприятия информации, постановки цели и выбор путей её достижения (ОК-1);

– способности использовать знания о современной естественнонаучной картине мира в образовательной и профессиональной деятельности, применять методы математической обработки информации, теоретического и экспериментального исследования (ОК-4);

– способности логически верно выстраивать устную и письменную речь (ОК-6);

– владения основами речевой профессиональной культуры (ОПК-3);

– способности нести ответственность за результаты своей профессиональной деятельности (ОПК-4);

– способности использовать возможности образовательной среды для формирования универсальных видов учебной деятельности и обеспечения качества учебно-воспитательного процесса (ПК-5);

– готовности к взаимодействию с учениками, родителями, коллегами, социальными партнерами (ПК-6);

– готовности использовать систематизированные теоретические и практические знания для определения и решения исследовательских задач в области образования (ПК-11);

– способности использовать в учебно-воспитательной деятельности основные методы научного исследования (ПК-13).

В результате освоения названной дисциплины студент должен

*Знать*:

основные способы помехоустойчивого кодирования (ПК), применяемые в дискретном и непрерывном канале связи и передачи данных для повышения помехоустойчивости и энергетической эффективности цифровых систем передачи информации (блоковые коды, непрерывные коды, каскадные коды).

*Уметь*:

грамотно выбрать помехоустойчивый код в заданном канале и реализовать алгоритм его декодирования с предварительной оценкой сложности процедуры декодирования (число операций, частота переключений) и логической схемы декодера.

*Владеть*:

методами расчета помехоустойчивости в дискретном и непрерывном канале, оценкой энергетической и частотной эффективности реализуемых способов помехоустойчивого кодирования.

Учебное пособие содержит как теоретический, так и практический материал для более глубокого изучения дисциплины «Прикладная математика».

# 1. КОРРЕКТИРУЮЩИЕ КОДЫ

## 1.1. Принцип обнаружения и исправления ошибок корректирующими кодами

1.1.1. *Коды с обнаружением и исправлением ошибок*

Прежде чем начать рассмотрение специальных корректирующих кодов, следует отметить, что любой код способен обнаруживать и исправлять ошибки, если не все кодовые слова (кодовые комбинации) этого кода используются для передачи сообщений. Нагляднее рассмотреть это на примере блочных кодов, у которых последовательность символов на выходе источника разбивается на блоки (кодовые слова, кодовые комбинации), содержащие одинаковое число символов $k$. При этом для двоичного кода ансамбль сообщений будет иметь объем $N_p=2^k$. При помехоустойчивом кодировании это множество из $N_p$ сообщений отображается на множество $N = 2^n$ возможных кодовых слов, такая процедура и называется помехоустойчивым кодированием дискретных сообщений ($n$ – число символов в кодовом слове после кодирования, иногда его называют длиной кодовых слов, или значностью кода).

В общем случае для блочного равномерного кода с основанием $m$ код имеет $N = m^n$ возможных кодовых слов. Используемые для передачи сообщений кодовые слова из множества $N_p < N$ называют разрешенными, остальные кодовые слова из множества

$$N_з = (N - N_p) \qquad (1.1)$$

не используются и называются запрещенными (неразрешенными для передачи).

Сущность обнаружения ошибок схематично поясняется на рис. 1.1, *а*. Если в результате искажений в канале связи переданное (разрешенное) кодовое слово $A_i$ ($i$ = 1,2, … $N_p$) превращается в одно из запрещённых $B_j$ ($j$ = 1,2, … $N_з$), то ошибка обнаруживается, так как

такое слово не могло быть передано. Ошибка не обнаруживается только в том случае, когда очередное передаваемое кодовое слово превращается в другое разрешенное, например $A_j$, которое также могло быть передано.

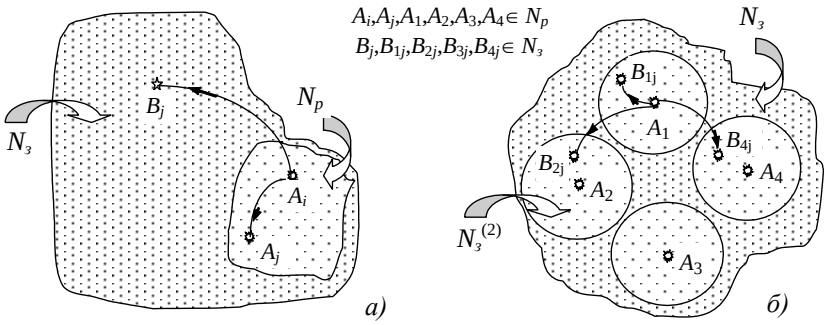



Рис. 1.1. Обнаружение и исправление ошибок

Исправление ошибок представляет собой более сложную операцию, так как кроме обнаружения наличия ошибки в принятом кодовом слове необходимо определить и местоположение искаженного кодового символа (а если $m > 2$, то и характер искажения). Для того чтобы рассматриваемый код исправлял ошибки, необходимо часть или всё множество $N_з$ запрещённых кодовых слов разбить на $N_p$ непересекающихся подмножеств $N_з^{(i)}$ ($i$ = 1,2, … $N_p$) по количеству разрешенных кодовых слов. Каждое из подмножеств $N_з^{(i)}$ в декодере приемника приписывается одному из разрешенных кодовых слов (рис. 1.1, *б*).

Способ приема заключается в том, что если принятое кодовое слово принадлежит подмножеству $N_з^{(i)}$, считается переданным $A_i$ разрешенное кодовое слово. Ошибка не может быть исправлена (исправляется неверно), если переданное кодовое слово $A_i$ в результате искажений превращается в кодовое слово любого другого подмножества $N_з^{(j)}$, ($j \neq i$). На рис. 1.1, *б* ошибка в запрещенном кодовом слове $B_{1j}$ будет исправлена, так как это слово принадлежит подмножеству $N_з^{(1)}$, приписанному к переданному разрешенному слову $A_1$; ошибка в кодовых словах $B_{2j}$ или $B_{4j}$ не будет исправлена, так как эти слова относятся к подмножествам, приписанным к другим разрешённым кодовым словам.

Если принятое кодовое слово попадает в подмножество запрещенных слов, не принадлежащих ни к одному из подмножеств $N_з^{(i)}$ ($i = 1,2, \ldots N_р$), то ошибка только обнаруживается, но не исправляется. Этот признак может быть использован для исправления ошибки другими методами, например методом переспроса.

Свойства кода по обнаружению и исправлению ошибок характеризуются количественно коэффициентами обнаружения $K_{об}$ и исправления ошибок $K_{ис}$, которые показывают, во сколько раз уменьшается вероятность ошибки после декодирования по сравнению с её величиной на входе приемного устройства (декодера) благодаря обнаружению ошибок или их исправлению соответственно. Ошибки в кодовых словах могут иметь произвольную конфигурацию, что определяется случайным характером помех в канале связи.

Число ошибочных символов в принятом кодовом слове называется кратностью ошибки $t$, при длине кодового слова из $n$ символов она изменяется в пределах от 0 до $n$.

$$\text{Если } P_n = P_{вх}(\geq 1, n) - \tag{1.2}$$

это вероятность ошибки кратности $t \geq 1$ в $n$ разрядном кодовом слове на входе декодера, а $P_{об}$ – вероятность обнаружения ошибок в декодере, то *коэффициент обнаружения* определяется следующим выражением:

$$K_{об} = \frac{P_{вх}(\geq 1, n)}{P_{вх}(\geq 1, n) - P_{об}}. \tag{1.3}$$

Коэффициент исправления ошибок будет определяться выражением

$$K_{ис} = \frac{P_{вх}(\geq 1, n)}{P_{вх}(\geq 1, n) - P_{ис}}, \tag{1.4}$$

где $P_{ис}$ – вероятность исправления ошибок в декодере.

Последняя численно равна вероятности ошибок в кодовом слове, кратность которых не превышает величины кратности гарантированно исправляемых ошибок $t_{ис}$, то есть $P_{ис} = P_{вх}(\leq t_{ис}, n)$.

Коэффициент исправления кода всегда меньше коэффициента обнаружения, что является общим условием для любых корректирующих кодов.

Для реализации потенциальных возможностей кода, исправляющего ошибки, необходимо учитывать статистический характер ошибок в реальных каналах связи, в которых предполагается применение этого кода. Разбиение неразрешенных комбинаций на подмножества $N_з^{(i)}$ должно выполняться таким образом, чтобы исправлялись ошибки, появление которых наиболее вероятно в данном канале связи.

В общем случае передаваемая кодовая комбинация искажается случайным образом, что определяется случайным характером помех в канале связи. В реальных системах связи при многообразии характера действующих в линии связи помех распределение кратностей ошибок в дискретном канале связи может быть самым различным. Поэтому построению декодера, исправляющего ошибки, должно предшествовать изучение статистических свойств канала связи. В качестве примера, на рис. 1.2 приведены кривые распределения кратностей ошибок $P_n(t)$ для двух случаев: для двоичного канала с независимыми ошибками в кодовых символах $p$ – кривая 1 (биномиальное распределение)

$$P_n(t) = C_n^t\, p^t(1-p)^{n-t} \qquad (1.5)$$

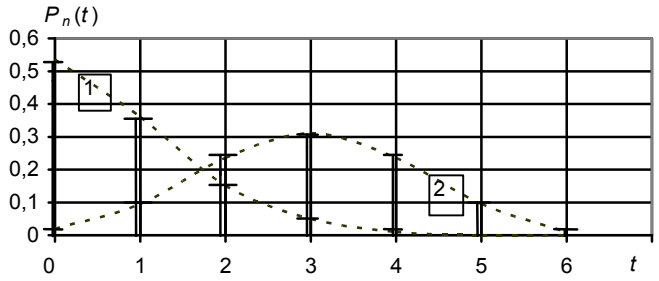



Рис. 1.2. Распределение кратностей ошибок

и кривая 2 для канала, в котором передаваемое кодовое слово с одинаковой вероятностью может превратиться в другое кодовое слово данного кода (из множества $N$).

$$P_n(t) = C_n^t / m^n \,. \qquad (1.6)$$

Графики соответствуют длине кодового слова $n = 6$.

Для приведенных на рис. 1.2 распределений кратностей ошибок определим величины коэффициентов исправления для корректиру-

ющего кода с параметрами $m=2$, $n=6$ ($N=2^6=64$), $N_p=10$, исправляющего одиночные ошибки: $t_{ис}=1$.

Вероятность ошибки в передаваемом кодовом слове в канале с распределением кратностей ошибок $P_n(t)$, соответствующим кривой 1 (см. рис. 1.2), и вероятностью искажения символа кода $p = 0{,}1$ равна

$$P_n = P_{вх}(\geq 1, n) = 1-(1-p)^n = 1-0{,}9^6 = 0{,}465.$$

Вероятность исправления ошибки (вероятность ошибки с кратностью $t=1$):

$$P_{ис} = C_n^t \cdot p^t (1-p)^{n-t} = C_6^1 \cdot p(1-p)^5 = 6 \cdot 0{,}1 \cdot 0{,}9^5 = 0{,}36.$$

Тогда

$$K_{ис} = \frac{P_{вх}(\geq 1, n)}{P_{вх}(\geq 1, n) - P_{ис}} = \frac{0{,}465}{0{,}465 - 0{,}36} \approx 4{,}4.$$

В канале связи с распределением кратностей ошибок $P_n(t)$, соответствующим кривой 2 (см. рис. 1.2), вероятность исправления ошибки (вероятность ошибки с кратностью $t=1$) равна

$$P_{ис} = P_n \cdot \frac{\sum_{i=1}^{t} C_n^t}{m^n} = P_n \frac{c_6^1}{2^6},$$

где $\sum_{i=1}^{t} C_n^t$ – доля ошибок, кратность которых $\leq t_{ис}$ из общего числа возможных ошибок $m^n$. Тогда коэффициент исправления равен

$$K_{ис} = \frac{P_n}{P_n - P_{ис}} = \frac{P_n \cdot}{P_n - P_n \frac{C_6^1}{2^6}} = \frac{2^6}{2^6 - C_6^1} = \frac{64}{64-6} \approx 1{,}24.$$

Таким образом, один и тот же код в первом случае исправляет примерно в четыре раза больше ошибок, чем во втором. Это объясняется тем, что в первом случае наибольшее количество ошибок имеет кратность $t$ = 1 и исправляется данным кодом, у которого каждому разрешенному кодовому слову приписывается подмножество ближайших неразрешенных слов, а во втором случае наибольшее количество ошибок имеет кратность $t$ >1, которые не исправляются данным кодом.

Очевидно, что если в канале связи преобладают ошибки большой кратности, целесообразно к разрешенным кодовым словам приписывать подмножество таких неразрешенных слов, которые удалены от данного разрешенного на расстояние, соответствующее этим ошибкам.

1.1.2. *Кодовое расстояние, избыточность кода*

Обнаруживающая и исправляющая способности корректирующих кодов тесно связаны с расстояниями между разрешенными кодовыми словами. Расстояние между любой парой кодовых слов $A_i$ и $A_j$ выражает различие между ними:

$$d_{ij} = \sum_{k=1}^{n} \left| x_{ik} - x_{jk} \right|, \qquad (1.7)$$

где $x_{ik}$, $x_{jk}$ – координаты кодовых слов $A_i, A_j$ в $n$-мерном неевклидовом пространстве $L_n$.

Если код является двоичным, под расстоянием между парой кодовых слов понимается количество символов, в которых они отличаются между собой. Оно определяется сложением двух этих слов по модулю 2 и равно числу единиц в этой сумме. Например

$$\begin{array}{ll} 101001 & -A_i \\ \oplus\ 011011 & -A_j \\ \hline 110010 & d_{ij}=3 \end{array}$$

Знак $\oplus$ означает сумму по модулю 2 (сложение по модулю два выполняется по правилу: $0\oplus 0=0,\ 0\oplus 1=1,\ 1\oplus 0=1,\ 1\oplus 1=0$).

Напомним, что геометрической моделью $n$-значного двоичного кода является $n$-мерный куб с ребром, равным единице, каждая вершина которого представляет одно из возможных кодовых слов. Расстояние между словами $d_{ij}$ равно числу ребер куба, отделяющих одну вершину от другой. Наименьшее расстояние между любой парой разрешенных слов данного кода называется *кодовым расстоянием*

$$d = \min d_{ij}. \qquad (1.8)$$

Так как кратность ошибки $t$ в геометрическом представлении является расстоянием между переданной комбинацией и искаженной, то для обнаружения ошибок кратности $t_{об}$ требуется кодовое расстояние

$$d \geq t_{об} + 1. \qquad (1.9)$$

Для исправления ошибок кратности $t_{ис}$ требуется кодовое расстояние

$$d \geq 2t_{ис} + 1. \qquad (1.10)$$

Это означает, что для исправления ошибок искаженное кодовое слово должно располагаться ближе всего к соответствующему правильному слову.

Для исправления стираний кратности $t_{ст}$ требуется кодовое расстояние

$$d \geq t_{ст} + 1, \qquad (1.11)$$

то есть для исправления стираний требуется такое же кодовое расстояние, как и для обнаружения ошибок.

Способность корректирующих кодов обнаруживать и исправлять ошибки (или стирания) определяется передачей дополнительной (избыточной) информации по каналу связи. Коэффициент избыточности в соответствии с теорией информации, как известно, равен

$$g = \frac{\log N - \log N_p}{\log N} = \frac{\log m^n - \log m^k}{\log m^n} = \frac{n-k}{n} = \frac{r}{n}, \qquad (1.12)$$

где $r$ – число избыточных кодовых символов в слове ($k + r = n$).

Для каналов с независимыми ошибками вероятность приёма кодового слова с ошибками определяется очевидным выражением вида

$$P_{вх}\ (\geq 1, n) = 1 - (1-p)^n, \qquad (1.13)$$

а вероятность обнаружения ошибки в принятом кодовом слове при декодировании равна

$$P_{об} = P_{вх}(\leq t_{об}, n) = \sum_{i=1}^{t_{об}} C_n^i p^i (1-p)^{n-i}. \qquad (1.14)$$

Тогда вероятность необнаружения ошибки при декодировании соответственно равна $P_{но} = P_{вх}(> t_{об}, n) = P_{вх}\ (\geq 1, n) - P_{об}$, то есть

$$P_{но} = P_{вх}(> t_{об}, n) = P_{вх}(\geq 1, n) - \sum_{i=1}^{t_{об}} C_n^i p^i (1-p)^{n-i}\ . \qquad (1.15)$$

Тогда коэффициент обнаружения можно определить следующим образом:

$$K_{об} = 2^r \frac{P_{вх}(\geq 1, n)}{P_{вх}(> t_{об}, n)} \quad . \tag{1.16}$$

В настоящее время известно большое число корректирующих кодов, отличающихся по помехоустойчивости и способам построения. Применение некоторых из них ограничивается сложностью технической реализации кодирующих и декодирующих устройств.

На рис. 1.3 приведена классификация наиболее часто используемых корректирующих кодов, в которой отмечаются только семейства кодов без подробной детализации.

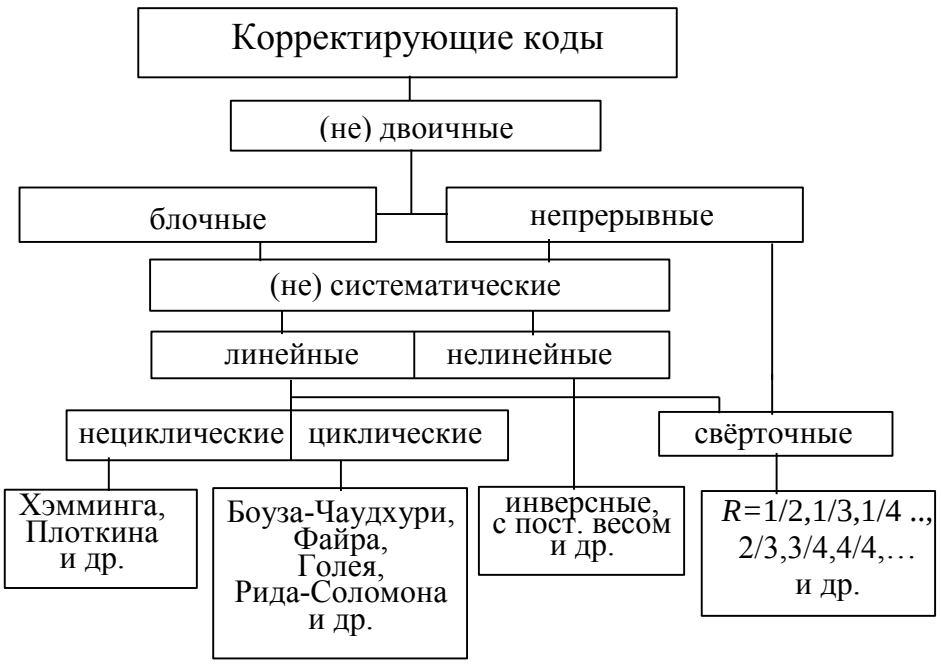



Рис. 1.3. Классификация корректирующих кодов

Применение корректирующих (помехоустойчивых) кодов является эффективным средством борьбы с ошибками в каналах связи с помехами. Такие коды используются либо только для обнаружения ошибок, либо для обнаружения и исправления ошибок (или ошибок и стираний в каналах со стиранием).

1.1.3. *Энергетический выигрыш кода*

В заключение рассмотрим энергетический выигрыш от помехоустойчивого кодирования для случая известных (заданных) параметров канала связи и кода. Вероятность ошибки на выходе дискретного канала связи $p_{\text{вых}}$ (или вероятность ошибки декодирования $p_д$) является функцией отношения сигнал/шум и качества используемого корректирующего кода. “Выигрыш от кодирования”, или “энергетический выигрыш” (ЭВК в децибеллах), который указывает на улучшение качества системы связи от использования данного способа кодирования или метода защиты от ошибок, определяется выражением

$$\text{ЭВК} = 10 \lg \frac{h_1^2}{a \cdot h_2^2} \text{ дБ}, \qquad (1.17)$$

где $h_1{}^2$, $h_2{}^2$ – отношения сигнал/шум в первой и второй сравниваемых системах связи при одинаковой вероятности ошибок на выходе;

$a$ – коэффициент, выравнивающий скорость передачи информации в сравниваемых системах.

Например, если первая система является системой без помехоустойчивого кодирования, а вторая – системой с обнаружением ошибок и переспросом, то $a = (n/k) \cdot T_{\text{ср}} / T$; здесь $T_{\text{ср}}$ – средняя длительность передачи кодового слова (или символа длительности $T$) в системе с переспросом. Для системы с кодом, исправляющим ошибки без переспроса, $a = n/k$.

Если снять ограничения на длину кодового слова и полосу частот, занимаемую системой связи, то предельный выигрыш от помехоустойчивого кодирования при данной вероятности ошибки в канале связи с гауссовским шумом будет равен [9]

$$10 \lg \frac{E / N_0}{0{,}693} = 10 \lg (E/N_0) + 1{,}6 \text{ дБ}. \qquad (1.18)$$

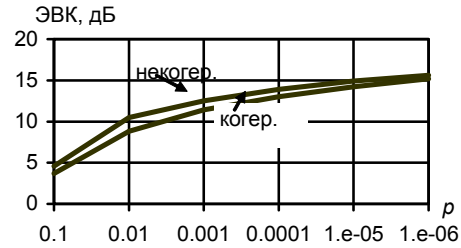



Рис. 1.4. Предельные значения ЭВК от кодирования при ЧМ

На рис. 1.4 приведены для примера кривые предельных значений ЭВК от кодирования при когерентном и некоге-

рентном приемах сигналов дискретной частотной модуляции (ЧМ) в зависимости от вероятности ошибки в дискретном канале связи.

В реальных системах связи длина кода и занимаемая полоса частот ограничены, для этих условий может быть определен асимптотический выигрыш для данного кода. Он зависит только от скорости кода $k/n$ и кодового расстояния.

Для каналов с жёстким решением (на выходе демодулятора двоичные символы 1 и 0)

$$\text{ЭВК} = 10\lg\left(g\,\frac{d+1}{2}\right)\text{дБ}, \tag{1.19}$$

для каналов с мягким решением (на выходе демодулятора многоуровневый сигнал)

$$\text{ЭВК} = 10\lg(g\cdot d)\text{дБ}. \tag{1.20}$$

Такой выигрыш достигается, когда $E/N_0 \to \infty$. Из этих соотношений видно, что мягкие решения обеспечивают дополнительный выигрыш не более 3 дБ (при $E/N_0 \to \infty$) и существенно меньше при реальных значениях отношения сигнал/шум.

## 1.2. Простейшие корректирующие коды

### 1.2.1. *Код с четным числом единиц*

Код с четным числом единиц является двоичным блочным кодом и образуется путем добавления к кодовому слову $k$-символьного кода одного избыточного символа так, чтобы количество единиц в новом $n$-символьном слове было четным. В таблице приведён пример кодирования пятизначного кода: $k = 5$, $n = 6$ .

*Таблица кодирования*

| $k = 5$ | | | | | $r = 1$ |
|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 | 6 |
| 1 | 0 | 1 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 | 1 | 0 |
| 0 | 0 | 0 | 0 | 1 | 1 |

Код обнаруживает все ошибки нечетной кратности. Обнаружение ошибок производится проверкой принятого кодового слова на четность, так как все разрешенные слова имеют четное число единиц, а неразрешенные – нечетное. Проверка на четность осуществляется суммированием всех символов слова по модулю два. Если слово имеет четное число единиц, то сумма его символов по модулю 2 равна 0.

Если в канале связи ошибки независимы и вероятность искажения кодового символа равна $p$, то согласно биномиальному закону распределения вероятность обнаружения ошибки равна

$$P_{об} = C_n^1 p(1-p)^{n-2} + C_n^3 p^3 (1-p)^{n-3} + C_n^5 p^5 (1-p)^{n-5} + \ldots; \qquad (1.21)$$

вероятность искажения кодового слова

$$P_n = 1-(1-p)^n;$$

вероятность необнаруженной ошибки

$$P_{но} = C_n^2 p^2 (1-p)^{n-2} + C_n^4 p^4 (1-p)^{n-4} + C_n^6 p^6 (1-p)^{n-4} + \ldots = P_n - P_o. \qquad (1.22)$$

Коэффициент избыточности этого кода $g = 1/n$.

### 1.2.2. *Код с постоянным весом*

Примером кода с постоянным весом является семизначный код с отношением единиц и нулей в каждом кодовом слове, равным $3/4$. Код имеет

$$N = \frac{n!}{m!(n-m)!} = \frac{7!}{3!\cdot 4!} = 35 \qquad (1.23)$$

разрешенных кодовых слов. Такого числа слов достаточно для помехоустойчивого кодирования всех кодовых слов 5-значного телеграфного кода.

Семизначный код 3/4 относится к неразделимым кодам с постоянным весом. В кодовом слове этого кода невозможно разделить символы на информационные и проверочные (избыточные). Обнаружение ошибок производится простым подсчетом единиц или нулей в принятом кодовом слове. Код обнаруживает все ошибки нечетной кратности и около 50 % ошибок четной кратности. Ошибки не обнаруживаются, если в одном кодовом слове искажается одинаковое число единиц и нулей. Например, если в разрешенном слове 1011000

искажены первый и второй (или второй и третий и т.д.) кодовые символы, то кодовое слово превращается в другое разрешенное – 0111000 и т.д.

Вероятность необнаруженной ошибки для кода 3/4 в канале связи с независимыми ошибками равна

$$P_{\text{нп}} = C_3^1 p(1-p)^2 \cdot C_4^1 p(1-p)^3 + C_3^2 p^2(1-p) \cdot C_4^2 p^2(1-p^2) +$$
$$+ C_3^3 p^3 \cdot C_4^3 p^3(1-p) == 12p^2(1-p)^5 + 18p^4(1-p)^3 + 4p^6(1-p).$$

Избыточность кода $g = r/n = 2/7 \approx 0{,}3.$ (1.24)

Обнаруживающая способность семизначного кода выше, чем шестизначного с проверкой на четность, но это достигается за счет увеличения избыточности.

## 1.3. Групповые коды

### 1.3.1. *Кодирование и декодирование групповых кодов*

Согласно теореме Шеннона для дискретных каналов с шумами скорость передачи информации может быть сколь угодно близкой к пропускной способности канала при сколь угодно малой вероятности ошибки. При этом не накладывается каких-либо ограничений на способ кодирования передаваемой информации и длину используемого кода.

Для построения кодов с хорошими характеристиками, которые обеспечивали бы малую вероятность ошибки на выходе канала связи при заданной избыточности, требуется очень большой набор кодовых слов, то есть кодовые слова такого кода должны содержать большое число кодовых символов $n$. При этом значительно усложняются анализ кода, а также процедура кодирования и декодирования. Например, если $n = 40$, то число возможных кодовых слов двоичного кода $N = 2^{40}$ и декодирующее устройство приемника должно помнить $2^{40} > 10^{12}$ кодовых слов. Возникает необходимость строить коды по определенным правилам, позволяющим осуществить достаточно просто операции кодирования и декодирования применяемого корректирующего кода.

В современной теории кодирования широко используются основные понятия высшей алгебры: множества, матрицы, векторные пространства, группы, кольца и поля [1, 2, 8, 9]. Групповые коды образуют особый класс кодов, построение которых производится по определенным правилам, известным из теории множеств. Из класса групповых кодов в дальнейшем будут рассматриваться только двоичные коды.

*Групповым* $(n,k)$ называется такой код, кодовые слова которого содержат $n$ символов; причем $k$ символов, расположенных в определенных местах каждого кодового слова, являются информационными, а остальные $(r=n-k)$ – проверочными (избыточными). Кодовые слова группового кода образуют группу относительно некоторой математической операции (оператора группы). Если оператор группы является линейным, то *групповой код также является линейным;* проверочные символы такого кода образуются в результате линейных операций с информационными символами. В случае двоичных кодов эти линейные операции сводятся к операции сложения по модулю 2. Кроме того, линейный код является *систематическим*, если последовательность информационных символов кода не изменяется в процессе кодирования.

Таким образом, групповым линейным кодом является код, кодовые слова которого образуют подгруппу конечной группы $G_n$. Конечной группой $G_n$ в данном случае называется множество кодовых слов, отличающееся тем свойством, что при сложении пары кодовых слов данного множества по модулю 2 образуются кодовые слова этого же множества, то есть если $A_1 \in G_n$ и $A_2 \in G_n$, то и

$$A_1 \oplus A_2 \in G_n . \qquad (1.25)$$

Порядок группы равен $2^n$, то есть равен числу кодовых слов в группе (числу элементов множества). Если для передачи информации используется $k$ разрядов из общего числа $n$, то количество кодовых слов в таком коде равно $2^k$ (подмножество группы $G_n$ порядка $2^n$). При этом подмножество группы $G_n$ называется подгруппой, если оно само по себе образует группу относительно операции сложения по

модулю 2. В подгруппу обязательно входит нулевое слово, все кодовые символы которого равны нулю (нулевой член множества $G_n$). Например, группа $G_4$, имеющая порядок $2^4$ содержит в себе все подгруппы всех других порядков от $2^0$ до $2^4$. Подгруппа $2^0$ состоит только из одного кодового слова вида 0000, а подгруппа $2^4$ есть сама группа $G_n$, состоящая из 16 кодовых слов.

Простейшим примером группового кода является рассмотренный в 3.2.1 код с четным числом единиц. Этот код образует группу $G_n$, а разрешенные кодовые слова образуют подгруппу порядка $2^{n-1}$, так как каждое кодовое слово имеет только один проверочный символ, который образуется в результате суммирования по модулю 2 всех информационных разрядов $a_n = a_1 \oplus a_2 \oplus \ldots \oplus a_{n-1}$.

Так как разрешенные кодовые слова группового линейного кода образуют подгруппу относительно операции сложения по модулю 2, то для определения всех кодовых слов подгруппы достаточно найти $k$ линейно-независимых кодовых слов (сумма этих слов по модулю 2 не должна равняться нулю). Остальные $2^k - (k+1)$ кодовых слов ( кроме нулевого слова) находятся сложением по модулю 2 во всевозможных сочетаниях этих известных кодовых слов, так как

$$\sum_{i=2}^{k} C_k^i = 2^k - (k+1). \qquad (1.26)$$

Для построения группового кода удобно пользоваться понятиями *производящая матрица и проверочная матрица*. Производящая матрица $n$-разрядного кода, имеющего $k$ информационных разрядов, имеет $n$ столбцов и $k$ строк. Чаще всего информационными разрядами являются первые $k$ разрядов (слева). Производящая матрица в канонической форме образуется путем дополнения $r = n - k$ столбцов к квадратной единичной матрице, содержащей $k$ строк и $k$ столбцов.

$$\left| k \begin{cases} 1000\ldots00\, b_{11}b_{12}\ldots b_{1r} \\ 0100\ldots00\, b_{21}b_{22}\ldots b_{2r} \\ 0010\ldots00\, b_{31}b_{32}\ldots b_{2r} \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots \\ 0000\ldots10\, b_{i1}b_{i2}\ldots b_{ir} \\ 0000\ldots01\, b_{k1}b_{k2}\ldots b_{kr} \end{cases} \right| . \qquad (1.27)$$

Код, определяемый этой матрицей, содержит $2^k - 1$ ненулевых разрешенных комбинаций. Благодаря тому что первые $k$ столбцов канонической производящей матрицы строго определены, для сравнения эквивалентности двух групповых $(n,k)$ кодов достаточно сравнить только их дополнения. Коды будут эквивалентны, если их дополнения могут быть приведены к одному и тому же виду. При этом коды будут иметь одинаковые распределения кодовых расстояний и корректирующую способность.

Проверочная матрица строится для определения алгоритма кодирования и декодирования данного группового кода. Каноническая форма проверочной матрицы записывается путем дополнения $k$ столбцов к единичной матрице $r \times r$ (дополнение приписывается слева от единичной матрицы).

$$\overbrace{\phantom{a_{11}a_{21}\ldots a_{k1}}}^{k} \quad \overbrace{\phantom{100\ldots0}}^{r}$$

$$\left[\begin{array}{ll} a_{11}a_{21}\ldots a_{k1} & 100\ldots0 \\ a_{12}a_{22}\ldots a_{k2} & 010\ldots0 \\ \ldots\ldots\ldots\ldots\ldots\ldots & \ldots\ldots \\ a_{1r}a_{2r}\ldots a_{kr} & 000\ldots1 \end{array}\right] \begin{array}{l} \\ \\ \\ \}\, r \end{array}. \qquad (1.28)$$

Рассмотрим пример.

Построить линейный код $n = 7$, обеспечивающий исправление одиночных ошибок.

Решение задачи

Для исправления одиночной ошибки необходимо кодовое расстояние $d \geq 2t_u + 1 = 3$.

Можно показать, что минимально необходимое число проверочных разрядов при данном кодовом расстоянии определяется из следующего соотношения:

$$2^r \geq 1 + \sum_{j=1}^{j=t_u} C_n^j, \quad 2^r \geq 1 + C^1{}_7 = 8, \text{ то есть } r \geq 3. \qquad (1.29)$$

Строим производящую матрицу, для этого берём квадратную единичную матрицу из $k = (n - r)$ строк и столбцов и путём перебора добавляем проверочные разряды так, чтобы расстояние между кодовыми словами было не меньше $d$. Четыре строки матрицы образуют 4 кодовых слова искомого кода.

| | |
|---|---|
| | 1000 011 |
| | 0100 110 |
| | 0010 101 |
| | 0001 111 |
| 1⊕2 | 1100 101 |
| 1⊕3 | 1010 110 |
| 1⊕4 | 1001 100 |
| 2⊕3 | 0110 011 |
| 2⊕4 | 0101 001 |
| 3⊕4 | 0011 010 |
| 1⊕2⊕3 | 1110 000 |
| 2⊕3⊕4 | 0111 100 |
| 1⊕3⊕4 | 1011 011 |
| 1⊕2⊕4 | 1101 010 |
| 1⊕2⊕3⊕4 | 1111 111 |

(1.30)

Остальные 11 кодовых слов из общего числа $2^k - 1$ находятся суммированием по модулю 2 всевозможных сочетаний строк матрицы. Нулевое слово 0000 000 обычно не используется, хотя и принадлежит данному коду (принадлежит к выбранной подгруппе группы $G_7$).

Как видно из (1.30), кодовое расстояние данного кода равно трём ($d = 3$), то есть такой код способен исправить любые одиночные ошибки.

Используя вычисленные кодовые слова, запишем проверочную матрицу

$$H=\begin{array}{c}\begin{matrix}1&2&3&4&5&6&7\end{matrix}\\ \begin{vmatrix}0&1&1&1&1&0&0\\1&1&0&1&0&1&0\\1&0&1&1&0&0&1\end{vmatrix}\end{array}.$$

Для проверочной матрицы выбраны такие кодовые слова из (1.30 ), проверочные символы которых образуют единичную матрицу, а число единиц является чётным; следовательно, все строки проверочной матрицы удовлетворяют проверкам на чётность

$$\begin{aligned} a_2 \oplus a_3 \oplus a_4 \oplus a_5 &= 0,\\ a_1 \oplus a_2 \oplus a_4 \oplus a_6 &= 0,\\ a_1 \oplus a_3 \oplus a_4 \oplus a_7 &= 0. \end{aligned} \qquad (1.31)$$

Полученное уравнение определяет правило проверки всех комбинаций данного кода в процессе их декодирования в приёмнике. Операция кодирования в передатчике, т.е. вычисление проверочных кодовых элементов определяется алгоритмом

$$a_5 = a_2 \oplus a_3 \oplus a_4,$$
$$a_6 = a_1 \oplus a_2 \oplus a_4,$$
$$a_7 = a_1 \oplus a_3 \oplus a_4. \qquad (1.32)$$

Если от источника сообщений в кодирующее устройство поступает последовательность вида 0110, то кодирующее устройство выдаёт комбинацию вида 0110011 в соответствии с записанным выше алгоритмом.

Допустим, что в канале связи комбинация была искажена и приняла вид 0100011. Декодирующее устройство осуществляет проверки на чётность в соответствии с уравнениями

$$a_2 \oplus a_3 \oplus a_4 \oplus a_5 = 1 \oplus 0 \oplus 0 \oplus 0 = 1,$$
$$a_1 \oplus a_2 \oplus a_4 \oplus a_6 = 0 \oplus 1 \oplus 0 \oplus 1 = 0,$$
$$a_1 \oplus a_3 \oplus a_4 \oplus a_7 = 0 \oplus 0 \oplus 0 \oplus 1 = 1. \qquad (1.33)$$

Наличие единиц в процессе проверок указывает на искажение кодового слова, а результаты проверок являются элементами синдрома ошибки $S(1,0,1)$.

Синдромом последовательности кодовых символов (синдромом ошибки) называется последовательность $S$, определяемая матричным равенством

$$\mathbf{S} = z \cdot \mathbf{H}^{\mathrm{T}}, \qquad (1.34)$$

где $\mathbf{H}^{\mathrm{T}}$ – транспонированная проверочная матрица кода.

Учитывая, что $z = a + e$ – декодируемая кодовая последовательность, $e$ – последовательность ошибок, $a \cdot \mathbf{H}^{\mathrm{T}} = 0$.

$$\mathbf{S} = z \cdot \mathbf{H}^{\mathrm{T}} = (a + e)\,\mathbf{H}^{\mathrm{T}} = e \cdot \mathbf{H}^{\mathrm{T}}. \qquad (1.35)$$

Если необходимо исправить ошибку, анализ элементов синдрома продолжается. На основании второй проверки делается заключение, что символы $a_1$, $a_2$, $a_4$, $a_6$ не искажены. Следовательно, искажённым является один из символов: $a_3$, $a_5$, $a_7$ (код позволяет исправить только одиночную ошибку). Так как символ $a_3$ входит и в первое, и во второе уравнения, то искажён именно он. Этот символ в принятом кодовом слове заменяется на противоположный. Получается кодовое слово 0110011, совпадающее с переданным.

Рассмотренный пример позволяет построить обобщенную структурную схему декодера линейного кода, показанную на рис. 1.5.

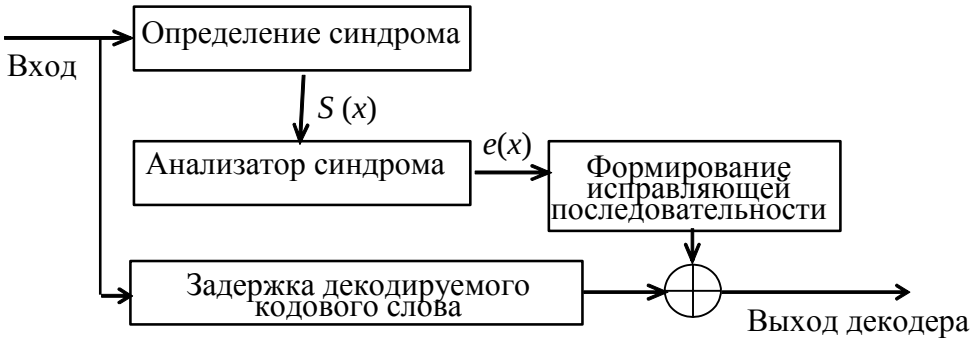



Рис. 1.5. Структурная схема декодера линейного кода

Наиболее сложным устройством декодера, определяющим возможность его реализации, является анализатор синдрома. Синдромы ошибок используемых на практике корректирующих кодов могут содержать десятки и сотни элементов, что затрудняет реализацию устройств логического анализа системы проверочных уравнений (1.33). Число логических операций, необходимое для декодирования кодового слова длиной $n$ (сложность декодера ), обычно экспоненциально увеличивается с ростом $n$. Поэтому усилия разработчиков в основном направлены на поиск таких корректирующих кодов и таких методов декодирования, которые упрощали бы процедуру анализа синдрома ошибки.

Существенное упрощение процедуры декодирования достигается при использовании кодов Хэмминга и циклических кодов.

### 1.3.2. *Коды Хэмминга*

*Кодами Хэмминга* обычно называют групповые коды:

– с кодовым расстоянием $d = 3$, исправляющие одиночные ошибки;

– с кодовым расстоянием $d = 4$, исправляющие одиночные ошибки и обнаруживающие тройные.

Коды второго вида получаются из кодов первого вида добавлением одного проверочного символа, равного сумме по модулю 2 всех остальных кодовых символов.

Проверочные символы кода Хэмминга, как и других систематических кодов, вычисляются путем сложения по модулю 2 информа-

ционных символов, расположенных в определенных разрядах кодового слова. При декодировании эти информационные символы вместе с соответствующим проверочным символом должны удовлетворять проверке на четность (сумма этих символов по модулю 2 должна равняться нулю).

Кроме этого общего требования, для кодов Хэмминга является обязательным, чтобы результат проверок на четность при декодировании искаженного кодового слова указывал также номер разряда, в котором расположен искаженный символ. Этот номер представляется числом, которое образуется при записи результатов 1-й, 2-й, ... $r$-й проверок в двоичной системе счисления ($b_r, b_{r-1}, .. b_2, b_1$).

Обозначим разряды кодового слова кода Хэмминга в виде последовательности $a_1,\ a_2,\ a_3, ... a_k, a_{k+1} ... a_n$, где 1, 2, 3,...$k$..., $n$ – натуральные числа, представляющие собой номера разрядов. Следовательно, первая проверка должна охватывать те нечётные символы, при записи номеров которых в двоичной системе счисления обязательно имеется единица в первом разряде двоичного числа, то есть $1 \to 1,\ 3 \to 11,\ 5 \to 101,\ 7 \to 111,\ 9 \to 1001$ и так далее;

$$a_1 \oplus a_3 \oplus a_5 \oplus a_7 \oplus a_9 ... = 0.$$

Вторая и последующие проверки строятся следующим образом:

$$\begin{aligned} a_2 \oplus a_3 \oplus a_5 \oplus a_7 \oplus a_{10} ... &= 0, \\ a_4 \oplus a_5 \oplus a_6 \oplus a_7 \oplus a_{12} ... &= 0, \\ a_6 \oplus a_9 \oplus a_{10} \oplus a_{11} \oplus a_{12} ... &= 0. \end{aligned} \qquad (1.36)$$

Проверочные символы располагаются в тех разрядах, которые участвуют только в одной проверке. Такими разрядами являются 1-й, 2-й, 4-й, 8-й, 16-й и так далее. Необходимое число проверочных разрядов в кодовом слове определяется выражением (1.29), которое для кодов Хэмминга ($d = 3$) удобнее использовать в виде $2^k \le {}^{2^n}\!/_{n}$.

Рассмотрим пример.

Необходимо закодировать кодом Хэмминга ($d = 3$) комбинацию из пяти информационных символов 10011.

Определяем необходимое число проверочных разрядов: $2^5 \le 2^n/n$ откуда $n \ge 8$. Принимаем $n = 9,\ r = n - k = 9 - 5 = 4$.

Проверочные символы должны занять 1, 2, 4 и 8-й разряды кода, а информационные 3(1), 5(0), 6(0), 7(1) и 9(1)-й разряды (в скобках указаны значения символов). Определяем значения проверочных символов согласно уравнениям (1.36):

$$a_1 = a_3 \oplus a_5 \oplus a_7 \oplus a_9 = 1 \oplus 0 \oplus 1 \oplus 1 = 1,$$

$$a_2 = a_3 \oplus a_6 = 1 \oplus 0 \oplus 1 = 0,$$

$$a_4 = a_5 \oplus a_6 \oplus a_7 = 0 \oplus 0 \oplus 1 = 1,$$

$$a_6 = a_9 = 1.$$

Таким образом, передаваемое кодовое слово кода Хэмминга имеет вид 101100111.

Пусть в канале связи символ, находящийся в 5-м разряде, был искажен и кодовое слово приняло вид 101110111. В приемнике в процессе декодирования производятся проверки согласно уравнениям (1.36):

$$a_1 \oplus a_3 \oplus a_5 \oplus a_7 \oplus a_9 = 1 \oplus 1 \oplus 1 \oplus 1 = 1 \to b_1,$$

$$a_2 \oplus a_3 \oplus a_6 \oplus a_7 = 0 \oplus 1 \oplus 0 \oplus 1 = 0 \to b_2,$$

$$a_4 \oplus a_5 \oplus a_6 \oplus a_7 = 1 \oplus 1 \oplus 0 \oplus 1 = 1 \to b_3,$$

$$a_8 \oplus a_9 = 1 \oplus 1 = 0 \to b_4.$$

Записываем результат проверки в виде $b_4 b_3 b_2 b_1 = 0101$, что равно десятичному числу 5, которое указывает номер искаженного разряда. Следовательно, в 5-м разряде необходимо изменить 1 на 0.

После исправления искаженного символа в информационных разрядах получим последовательность символов 10011, которая совпадает с переданной комбинацией информационных символов.

**Пример.** В примере предыдущего подпункта проверочная матрица была выбрана в виде

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

Если при передаче кодового слова $(10011)$ произошла ошибка в первом бите, т.е. $Y$ = (00011), то синдром $S = Y \cdot \mathbf{H}^{\mathrm{T}}$ = 11 однозначно укажет на номер столбца матрицы **H**. Если же ошибка произошла в четвертом бите, т.е. $Y$ = (10001), то $S$ = (10), но таких столбцов в матрице **H** два!

**Вывод.** Для однозначного обнаружения места ошибки достаточно, чтобы все столбцы матрицы **H** были различными.

**Пример.** Для проверочной матрицы (7, 4)-кода Хэмминга

$$\mathbf{H}_{3\times 7} = \begin{pmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 1 & 0 & 0 & 1 \end{pmatrix},$$

вектор $X$ = (0011110) является кодовым. Если при передаче произошла лишь одна ошибка и на выходе канала получили вектор $Y$ = (1011110), то синдром этого вектора $S = Y\mathbf{H}^{\mathrm{T}}$ = (111). Поскольку $S^{\mathrm{T}}$ совпадает с первым столбцом матрицы **H**, то заключаем, что ошибка произошла в первом разряде. Тут же исправляем его на противоположный: $X = Y + \mathbf{e}_1$.

Если при передаче произошли две ошибки и на выходе канала получили вектор $\hat{Y}$ = (1011010), то синдром этого вектора $S = Y\mathbf{H}^{\mathrm{T}}$ = (011). Поскольку синдром ненулевой, то факт наличия ошибки подтвержден. Однако корректно исправить ее – по аналогии с предыдущим – не удается. $S^{\mathrm{T}}$ совпадает с третьим столбцом матрицы **H**, но в третьем разряде полученного вектора ошибки нет.

Наконец, если при передаче произошли три ошибки и на выходе канала получили вектор $\hat{Y}$ = (1011011), то синдром этого вектора $S = Y\,\mathbf{H}^{\mathrm{T}}$ = (010). Наличие ошибки подтверждено, исправление невозможно. Если же при передаче того же вектора $X$ = (0011110) получаем вектор $\hat{Y}$ = (1111111) (также с тремя ошибками), то его синдром оказывается нулевым: $\hat{Y}\,\mathbf{H}^{\mathrm{T}}$ = (000) и ошибка не обнаруживается.

**Пример.** Для (7,4)-кода Хэмминга матрицу **H**, построенную в предыдущем примере, переупорядочим по столбцам; будем рассматривать ее в виде

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow & \uparrow \\ 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{pmatrix}$$

Распишем проверочные соотношения $X\mathbf{H}^{\mathrm{T}} = 0$ покомпонентно:

$$\begin{cases} x_4 + x_5 + x_6 + x_7 = 0 \\ x_2 + x_3 + x_6 + x_7 = 0 \\ x_1 + x_3 + x_5 + x_7 = 0 \end{cases} \Leftrightarrow \begin{cases} x_1 + x_3 + x_5 + x_7 = 0 \\ x_2 + x_3 + x_6 + x_7 = 0 \\ x_4 + x_5 + x_6 + x_7 = 0 \end{cases}$$

Переписанные в последнем виде эти уравнения представляют конечный пункт прямого хода метода Гаусса решения системы линейных уравнений, а именно трапециевидную форму этой системы. Если бы мы поставили задачу поиска общего решения этой (однородной) системы и нахождения фундаментальной системы решений, то в качестве зависимых переменных однозначно бы выбрали $x_1$, $x_2$, $x_3$. Выпишем это общее решение:

$$x_1 = x_3 + x_5 + x_7,\ x_2 = x_3 + x_6 + x_7,\ x_4 = x_5 + x_6 + x_7\ .$$

Это и есть проверочные соотношения, а проверочными разрядами кодового вектора являются 1-й, 2-й и 4-й.

Проверим правильность этих рассуждений. Придадим оставшимся разрядам произвольные значения, например: $x_3 = 1$, $x_5 = 1$, $x_6 = 0$, $x_7 = 1$. Тогда $x_1 = 1$, $x_2 = 0$, $x_4 = 0$ и кодовый вектор $X = (1010101)$. Пусть на выходе из канала он превратился в $Y = (1000101)$. Синдром этого вектора $Y\ \mathbf{H}^{\mathrm{T}} = (011)$ – это двоичное представление числа 3. И ведь действительно, ошибка в третьем разряде!

**Пример.** Для (7,4)-кода Хэмминга проверочная матрица в упорядоченном виде имеет вид

$$H_{(7,4)} = \begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{vmatrix}.$$

Пусть переданное кодовое слово $\upsilon(1,0) = 1101001$, а принятое – $\upsilon(1,0) = 1101101$. Синдром, соответствующий принятому слову, будет равен

$$S(1,0) = \upsilon'(1,0) \cdot H^{T}_{(7,4)} = [1101101] \cdot \begin{vmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 1 & 1 & 1 \end{vmatrix} = [101].$$

Вычисленный синдром указывает на ошибку в пятой позиции. Проверочная матрица в упорядоченном виде представляет совокупность проверочных уравнений, в которых проверочные символы занимают позиции с номерами $2^i$($i$ = 0,1,2...). Для (7,4)-кода Хэмминга

$$\upsilon_1 = \upsilon_3 + \upsilon_5 + \upsilon_7;$$
$$\upsilon_2 = \upsilon_3 + \upsilon_6 + \upsilon_7;$$
$$\upsilon_4 = \upsilon_5 + \upsilon_6 + \upsilon_7,$$

проверочными уравнениями будут, где $\upsilon_1$, $\upsilon_3$ и $\upsilon_4$ – проверочные символы. Элементы синдрома определяются из выражений

$$S_0 = \upsilon_1 + \upsilon_3 + \upsilon_5 + \upsilon_7;$$
$$S_1 = \upsilon_2 + \upsilon_3 + \upsilon_6 + \upsilon_7;$$
$$S_2 = \upsilon_4 + \upsilon_5 + \upsilon_6 + \upsilon_7.$$

Корректирующая способность кода Хэмминга может быть увеличена введением дополнительной проверки на четность. В этом случае проверочная матрица для рассмотренного (7,4)-кода будет иметь вид

$$H_{(8,4)} = \begin{vmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{vmatrix},$$

а кодовое расстояние кода $d_0$=4. Проверочные уравнения используются для построения кодера, а синдромные – декодера кода Хэмминга.

# 2. ЦИКЛИЧЕСКИЕ КОДЫ

## 2.1. Кодирование циклических кодов

Среди множества кодовых слов группы $G_n$ порядка $2^n$ можно найти такое подмножество (подгруппу группы $G_n$ ), у которого все строки производящей матрицы образуются путем умножения некоторого многочлена $g(x)$ степени $(n-k)$ последовательно на $1, x, x^2 ... x^{k-1}$. Такие подгруппы называются циклическими [1, 2]. Следовательно, для определения циклической подгруппы достаточно знать некоторый производящий многочлен $g(x)$, степень которого равна числу проверочных символов $r = n - k$. Все остальные строки производящей матрицы связаны с производящим многочленом жесткой зависимостью. Кодовые слова, не входящие непосредственно в производящую матрицу, образуются, как и в любом систематическом коде, суммированием строк производящей матрицы по модулю 2 во всевозможных сочетаниях. Все кодовые слова циклического кода делятся без остатка на производящий многочлен $g(x)$, так как этот многочлен входит в каждое слово в качестве сомножителя. Частное от деления образует многочлен степени $(k-1)$, представляющий собой запись в виде многочлена информационных элементов данного кодового слова.

Для построения проверочной матрицы циклического кода необходимо найти многочлен

$$h(x) = \frac{1 + x^n}{g(x)}, \qquad (2.1)$$

который образует первую строку проверочной матрицы. Остальные ее строки находятся умножением многочлена $h(x)$ последовательно на $x, x^2, ... x^{k-1}$.

Таким образом, циклическими называются групповые коды, образуемые путем умножения каждого кодового слова *k*-символьного кода, выраженного в виде многочлена $Q(x)$, на некоторый произво-

дящий многочлен $g(x)$ степени $(n-k)$. Поэтому в теории циклических кодов кодовые слова часто представляют в виде многочленов $a_{n-1}x^{n-1}+a_{n-2}x^{n-2}+\ldots+a_2x^2+a_1x+a_0$, где коэффициенты $a_i$ принимают значения либо 0, либо 1.

Умножение многочленов производится по обычным правилам алгебры, но с приведением подобных членов по модулю 2.

Например, умножение в виде многочленов в двоичном виде

$$\begin{array}{r} \times\ x^3+x+1 \\ \underline{\quad x^2+1} \\ x^3+0+x+1 \\ \oplus\quad x^5+0+x^3+x^2 \\ \hline x^5+0+0+x^2+x+1=x^5+x^2+x+1 \end{array} \qquad \begin{array}{r} \times\ 1011 \\ \underline{\quad 101} \\ 1011 \\ \oplus\quad 1011 \\ \hline 100111 \end{array} \qquad (2.2)$$

Деление многочленов производится также с учетом операции сложения по модулю 2.

$$\begin{array}{r|l} \oplus\ x^5+0+0+x^2+x+1 & \underline{\quad x^2+1} \\ \underline{x^5+0+x^3} & x^3+x+1 \\ \oplus\ x^3+x^2+x+1 & \\ \underline{x^3+0+x} & \\ \oplus\ x^2+0+1 & \\ \underline{x^2+0+1} & \\ 0+0+0 & \end{array} \qquad \begin{array}{r|l} \oplus\ 100111 & \underline{\quad 101} \\ \underline{101} & 1011 \\ \oplus\ 111 & \\ \underline{101} & \\ \oplus\ 101 & \\ \underline{101} & \\ 000 & \end{array} \qquad (2.3)$$

Из способа построения циклического кода следует, что при циклическом сдвиге символов слова циклического кода образуется другое кодовое слово, принадлежащее данному коду (образуется другое разрешенное слово). Например, если кодовое слово 0001011 принадлежит данному циклическому коду, то и слово 0010110 также принадлежит этому коду, так как оно может быть получено умножением первого слова на $x$.

Для того чтобы построить циклический код, удовлетворяющий определенным требованиям (исправляющий или обнаруживающий ошибки заданной кратности), необходимо соответствующим образом выбрать производящий многочлен. Если код предназначен для исправления и обнаружения независимых ошибок, то необходимо обеспечить соответствующее кодовое расстояние между кодовыми словами циклического кода. Лучшими кодами, обеспечивающими эффективную борьбу с независимыми ошибками, являются циклические *ко-*

*ды Боуза – Чоудхури*. Производящий многочлен для этих кодов находится по заданному кодовому расстоянию и числу символов в кодовом слове.

Необходимое количество проверочных разрядов определяется выражением

$$r \geq \frac{(d-1)M}{2}, \tag{2.4}$$

где число $M$ определяется из соотношения

$$n = 2^M - 1. \tag{2.5}$$

Тогда производящий многочлен представляет собой наименьшее общее кратное (НОК) произведение неприводимых многочленов $M_i(x)$, где $i = 1,3,5,..d-2$ – порядок многочлена

$$g(x) = \text{НОК}[M_1(x) \cdot M_3(x) \ldots M_{r-2}]. \tag{2.6}$$

Неприводимым называется многочлен, не делящийся ни на какой многочлен, степень которого меньше $M$.

Некоторые неприводимые многочлены приведены в табл. 2.1. Более подробные таблицы можно найти в специальной литературе, например в [1]. Широко распространены также машинные методы отбора производящих многочленов.

Табл. 2.1 *Неприводимые многочлены над GF(2)*

| Степень | Многочлен | Степень | Многочлен |
|---|---|---|---|
| 2 | $x^2 + x + 1$ | 16 | $x^{16} + x^{12} + x^3 + x + 1$ |
| 3 | $x^3 + x + 1$ | 17 | $x^{17} + x^3 + 1$ |
| 4 | $x^4 + x + 1$ | 18 | $x^{18} + x^7 + 1$ |
| 5 | $x^5 + x^2 + 1$ | 19 | $x^{19} + x^5 + x^2 + x + 1$ |
| 6 | $x^6 + x + 1$ | 20 | $x^{20} + x^3 + 1$ |
| 7 | $x^7 + x^3 + 1$ | 21 | $x^{21} + x^2 + 1$ |
| 8 | $x^8 + x^4 + x^3 + x^2 + 1$ | 22 | $x^{22} + x + 1$ |
| 9 | $x^9 + x^4 + 1$ | 23 | $x^{23} + x^5 + 1$ |
| 10 | $x^{10} + x^3 + 1$ | 24 | $x^{24} + x^7 + x^2 + x + 1$ |
| 11 | $x^{11} + x^2 + 1$ | 25 | $x^{25} + x^3 + 1$ |
| 12 | $x^{12} + x^6 + x^4 + x + 1$ | 26 | $x^{26} + x^6 + x^2 + x + 1$ |
| 13 | $x^{13} + x^4 + x^3 + x + 1$ | 27 | $x^{27} + x^5 + x^2 + x + 1$ |
| 14 | $x^{14} + x^{10} + x^6 + x + 1$ | 28 | $x^{28} + x^3 + 1$ |
| 15 | $x^{15} + x + 1$ | | |

## 2.2. Декодирование циклических кодов

При декодировании кодового слова циклического кода также производится его деление на производящий многочлен. Неискаженное кодовое слово должно делиться на производящий многочлен без остатка. Наличие остатка указывает на искажение слова.

Для определения искаженного символа кода и его исправления необходимо с полученным остатком произвести еще ряд операций. Иногда исправляются только одиночные ошибки, так как с увеличением кратности исправляемых ошибок резко усложняется декодирующее устройство, что затрудняет его реализацию.

Операции деления кодовых слов на производящий многочлен в процессе кодирования и декодирования производятся с помощью регистров сдвига с обратными связями, которые располагаются в соответствии с заданным производящим многочленом и осуществляются через сумматоры по модулю 2.

Как уже отмечалось, коды Боуза – Чоудхури используются в каналах с независимыми ошибками. Эти же коды способны обнаруживать пакеты ошибок, если их длина не превышает числа проверочных символов кода (под пакетом ошибок понимается группа кодовых символов, ошибки внутри которой располагаются на расстоянии, не превышающем определенного числа кодовых символов). Однако для обнаружения и исправления пакетов ошибок лучше использовать специальные коды, которые наиболее эффективны в каналах связи с группирующимися (коррелированными) ошибками. К числу таких кодов относятся циклические коды Файра. Длина пакета обнаруживаемых таким кодом ошибок значительно превышает число проверочных символов кодового слова. Кодирование и декодирование кодов Файра осуществляется также с помощью регистров сдвига с обратными связями.

Эффективность циклических кодов, как правило, увеличивается с увеличением длины кодовых слов. Поэтому в современной практике иногда применяются циклические коды, длина кодовых слов которых $n > 10^3$.

Рассмотрим пример.

Необходимо построить код Боуза – Чоудхури при $n = 7$ для исправления независимых ошибок кратности $t = 1$.

Для исправления независимых ошибок кратности $t = 1$ требуется кодовое расстояние $d \geq 2t + 1 = 3$. Находим число $M$: $n = 2^M - 1$; $7 = 2^M$; $M = 3$.

Число проверочных символов

$$r \geq \frac{(d-1)M}{2};\ r = \frac{(3-1)3}{2} = 3.$$

Находим производящий многочлен, учитывая, что из (2.6)

$$g(x) = \text{НОК}[M_1(x) \cdot M_3(x) \ldots M_{r-2}],\quad i = 1, 3, 5 \ldots d-2.$$

Тогда $i = d - 2 = 3 - 2 = 1$; $g(x) = M_1(x) = x^3 + x + 1 \rightarrow 1011$ и проверочный многочлен

$$h(x) = \frac{1 + x^7}{x^3 + x + 1} = x^4 + x^2 + x + 1 \rightarrow 10111. \qquad (2.7)$$

Строим производящую матрицу, которая образуется добавлением $n$-$k$ нулей к производящему многочлену, записанному в виде двоичного числа; остальные строки матрицы представляют собой циклический сдвиг первой строки. Суммируя по модулю 2 во всевозможных сочетаниях строки производящей матрицы, получим остальные 11 разрешённых кодовых слов (из общего числа $2^k - 1 = 2^4 - 1 = 15$, табл. 2.2).

$$\begin{array}{r\|\|l\|l} g(x) & 0001011 & 1 \\ x \cdot g(x) & 0010110 & 2 \\ x^2 \cdot g(x) & 0101100 & 3 \\ x^3 \cdot g(x) & 1011000 & 4 \end{array} \qquad (2.8)$$

Табл. 2.2. *Таблица кодовых слов*

| Кодовые слова | | Кодовые слова | |
|---|---|---|---|
| информационные | проверочные | информационные | проверочные |
| 0001 | 011 | 1001 | 011 |
| 0010 | 110 | 1110 | 100 |
| 0101 | 100 | 1000 | 101 |
| 1011 | 000 | 0110 | 001 |
| 0011 | 101 | 1111 | 111 |
| 0100 | 111 | 1100 | 010 |
| 1010 | 011 | 1101 | 001 |
| 0111 | 010 | | |

Нетрудно убедиться, что расстояние между любой парой кодовых слов в табл. 2.2 не меньше трёх. Следовательно, код может исправлять любые одиночные ошибки.

При кодировании и декодировании циклических кодов используется тот факт, что любое кодовое слово данного кода делится без остатка на производящий многочлен. Поэтому при кодировании к последовательности информационных символов добавляется справа $r$ нулей и получившаяся последовательность элементов делится на производящий многочлен. Остаток от деления представляет последовательность проверочных символов, которая передаётся в канал связи после передачи информационных символов.

Пусть, например, последовательность информационных символов имеет вид 0110.

| $k$ | $r$ | $g(x)$ |
|---|---|---|
| 0110000 | 1011 | |
| ⊕1011 | 0111 | |
| 01110 | | |
| ⊕ 1011 | | |
| 1010 | | |
| ⊕ 1011 | | |
| 001 | остаток | |

В результате образуется и передаётся кодовое слово 0110001 (см. табл. 2.2, кодовое слово 12).

При декодировании кодового слова циклического кода в приёмнике также производится его деление на производящий многочлен. Неискажённое слово должно делиться на производящий многочлен без остатка. Наличие остатка указывает на искажение.

Например,

| | |
|---|---|
| 0110001 | 1011 |
| ⊕1011 | 0111 |
| 01110 | |
| ⊕ 1011 | |
| 1011 | |
| ⊕ 1011 | |
| 0000 | |

а) принятое кодовое слово без ошибок (в остатке все нули),

$$
\begin{array}{l|l}
111001 & 1011 \\
\oplus\ 1011 & 10110 \\
\hline
01010 & \\
\oplus\ \ 1011 & \\
\hline
0011 & \\
\oplus\ \ 0000 & \\
\hline
011 \rightarrow S_i(x) &
\end{array}
$$

Синдром ошибки.

б) принятое кодовое слово с ошибками (в остатке 011). Для определения искажённого символа кода и его исправления необходимо с полученным остатком произвести ещё ряд операций, которые будут рассмотрены ниже.

***Оптимальные методы декодирования***

К оптимальным в том смысле, что они минимизируют среднюю вероятность ошибки

$$P_{\partial} = \sum_{a_i \in A} p(a_i)\, p(e / a_i), \qquad (2.9)$$

относятся методы декодирования по максимуму апостериорной вероятности (МАВ) и по максимуму правдоподобия (МП). Выбор декодированного варианта принятой последовательности $a*$ производится для первого метода по максимальному значению апостериорной вероятности $p(a_i / x_j)$, а для второго – по максимуму функции правдоподобия $p(x_j / a_i)$ по всем $a_i \in A$, где $a_i$ – передаваемая кодовая последовательность, $A$ – множество всех передаваемых последовательностей (алфавит), $x_i = a_i + e$ – принимаемая кодовая последовательность, $e$ – последовательность ошибок, $a*$ – декодированный вариант принятой последовательности.

Метод МП является частным случаем МАВ при равенстве априорных вероятностей передачи $p(a_i)$. Для симметричного канала связи оба эти метода декодирования эквивалентны выбору последовательности $a*$, отличающейся от принятой $x_i$ наименьшим числом символов. Это утверждение положено в основу реализации описанных выше оптимальных методов декодирования.

Трудности реализации оптимальных методов декодирования связаны с тем, что декодер должен иметь большую память, равную (или близкую) по объёму числу кодовых слов используемого кода. Поэтому оптимальные методы, как правило, используются для исправления ошибок малой кратности (одиночных ошибок или одиночных пакетов ошибок): это *метод стандартной расстановки* [1, 2] и *метод, основанный на выборе синдрома*.

Сущность метода, основанного на выборе синдрома, заключается в следующем.

До декодирования вычисляются все синдромы ошибок $S_i$ ($x$), исправляемых данным кодом, и сводятся в таблицу вместе с конфигурациями соответствующих ошибок $e_i(x)$ , $i = 1\ 2^{n-k}$.

Вычисляется синдром ошибки $S_j$ ($x$) декодируемого кодового слова.

Пошаговым сравнением синдрома $S_j$ ($x$) с синдромами $S_i$ ($x$), хранящимися в таблице, находится конфигурация ошибки $e_j(x)$, соответствующая синдрому $S_j$ ($x$).

Производится исправление ошибки в принятом кодовом слове суммированием (по модулю 2) этого слова с последовательностью ошибок $e_j(x)$.

Сложность этой процедуры связана с размерами таблицы, состоящей из $2^{n-k}$ строк длины $n$. Используя свойства циклических кодов и метод стандартной расстановки, можно несколько уменьшить объём таблицы и ускорить сравнение; однако и в этом случае эти методы практически можно использовать только для коротких кодов или для исправления одиночных ошибок.

Декодер Меггита представляет собой синдромный декодер, исправляющий одиночные ошибки, в памяти которого с целью упрощения хранится только один синдром ошибки: $S_{15}(x) = x^3+1$ (соответствует конфигурации ошибки $e_{15}(x) = x^{14}$ ). Синдромы остальных одиночных ошибок циклически сдвигаются в регистре синдрома до совпадения с $S_{15}(x)$; число циклов сдвига $i$ ( $i$ = 0,1,2...14) плюс единица равно номеру искаженного кодового символа.

Структурная схема декодера показана на рис. 2.1. Декодер работает следующим образом. Кодовое слово (с ошибками или без них) в виде последовательности из 15 двоичных символов поступает в буферный регистр и одновременно в регистр синдрома, где производится деление этого слова на производящий многочлен кода $g(x) = x^4 + x + 1$, в результате вычисляется синдром ошибки $S_j(x)$: $S_{0j}$, $S_{1j}$ , $S_{2j}$ , $S_{3j}$ − элементы синдрома. Ошибка обнаруживается, если хотя бы один элемент синдрома не равен нулю.

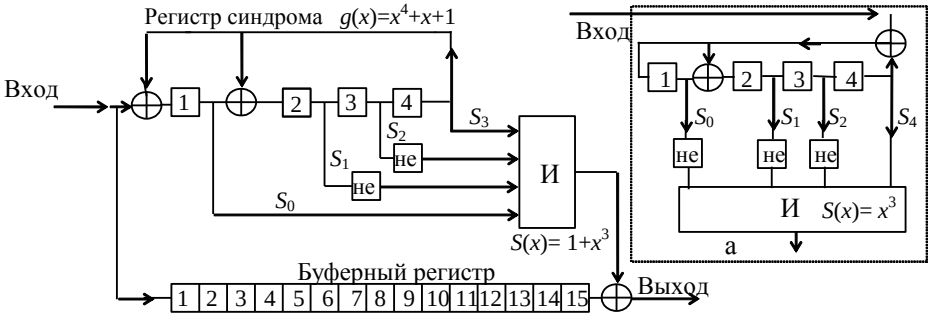



Рис. 2.1. Декодер Меггита циклического кода (15, 11)

Исправление ошибок производится в следующих 15 циклах сдвига. Если $S_j(x) = S_{15}(x)$, то ошибка в первом символе кодового слова, который находится в 15-й ячейке буферного регистра. Тогда в первом цикле схема выдаёт единицу и в сумматоре по модулю 2 на выходе буферного регистра корректируется первый символ кодового слова. Если ошибка в другом символе, то производится циклический сдвиг синдрома $S_j(x)$ в регистре синдрома по цепи обратной связи с учетом того, что вход декодера на циклах исправления ошибок отключен. В каждом $i$-м цикле проверяется равенство $S_{j+i}(x) = S_{15}(x)$ и в благоприятном случае на выходе схемы появляется импульс коррекции ошибки, инвертирующий символ на выходе буферного регистра.

В пунктирном квадрате „а” рис. 2.1. показана возможная модификация регистра синдрома, упрощающая реализацию схемы. Для этого принимаемая последовательность до ввода в регистр синдрома умножается на $x^4$, тогда синдром ошибки в первом символе кодового слова будет равен $S_{15}(x) = x^3$.

Для исправления ошибок большой кратности, как правило, используются *методы субоптимального декодирования*, которые позволяют упростить процедуру декодирования при незначительном

снижении качества. К субоптимальным методам декодирования относятся перестановочные методы, алгебраические (прямой, поэтапный, мажоритарный и др.) и пр.

Перестановочные методы декодирования основываются на свойстве симметрии линейных кодов, то есть на существовании такого множества перестановок символов кодовой последовательности, которые переводят ее в последовательность, принадлежащую тому же коду. Производя перестановки в декодируемой кодовой последовательности, можно найти такую перестановку $B_i$, в результате которой на информационных позициях будут находиться только неискаженные символы. Затем по известной процедуре кодирования $L$ определяются проверочные символы и тем самым исправляются искажённые символы. Выполнив обратную перестановку $B_i^{-1}$, получим исправленную кодовую последовательность.

Наибольшее распространение получила модификация перестановочного метода декодирования, предложенная Касами и Рудольфом. Отличие этого метода в том, что часть искажённых символов в результате циклической перестановки попадает на проверочные разряды, а искажённые символы, оставшиеся на информационных разрядах, исправляются с помощью покрывающих последовательностей.

Последовательность символов покрывает подмножество ошибок, если для любой последовательности ошибок $e(x)$ из этого подмножества можно найти такой её циклический сдвиг $B_i e$, чтобы на информационных позициях сдвинутая последовательность ошибок совпадала с покрывающей последовательностью. С помощью одной покрывающей последовательности можно исправить лишь часть ошибок, с ростом длины кода и кратности исправляемых ошибок увеличивается и количество покрывающих последовательностей. Момент совпадения покрывающей последовательности и циклически сдвинутой последовательности ошибок определяется по выполнению неравенства

$$G[S_z(x) - S_j(x)] \le 0{,}5(d - 1) - G[D_j(x)], \qquad (2.10)$$

где $G[*]$ – вес последовательности $[*]$, $S_j(x)$ – синдром $j$-й покрывающей последовательности; $S_z(x)$ – синдром принимаемой кодовой последовательности.

Для декодирования по алгоритму Касами – Рудольфа надо производить циклические сдвиги принимаемого кодового слова, вычислять вес $G[S_z(x) - S_j(x)]$ для каждой покрывающей последовательности. Если неравенство (2.10) выполнилось после $i$-го сдвига для $j$-й покрывающей последовательности, то декодированным вариантом принятого кодового слова следует считать последовательность $B_{n-i} \cdot L(B_i z - D_j)$.

Структурная схема декодера Касами – Рудольфа для кода Голея (23,12) приведена на рис. 2.2. Производящий многочлен этого кода

$$g(x) = x^{11} + x^9 + x^7 + x^6 + x^5 + x + 1;$$

множество ошибок, кратность которых не превышает трёх, покрывается тремя последовательностями ошибок $e_1(x) = 0$, $e_{17}(x) = x^{16}$, $e_{18}(x) = x^{17}$, имеющих синдромы:

$$\begin{aligned} S_1(x) &= 0; \\ S_{17}(x) &= x^8 + x^7 + x^4 + x^3 + x + 1; \\ S_{18}(x) &= x^9 + x^8 + x^5 + x^4 + x^2 + x. \end{aligned} \qquad (2.11)$$

Декодер отслеживает синдром ошибок декодируемого кодового слова, отличающийся от $S_1(x)$ не более чем в трёх позициях, а также синдромы ошибок, отличающиеся от $S_{17}(x)$ и $S_{18}(x)$ не более чем в двух позициях.

Декодирование производится в течение двух циклов. В первом цикле в течение 23 тактов производится запись принятого кодового слова в буферный регистр ($п_1$=0) и вычисление синдрома ошибки в синдромном регистре ($п_2$=0). Во втором цикле ($п_1$=1) из 23 тактов производится поиск и исправление ошибок путем циклического сдвига синдрома ошибки и его сравнения с покрывающими синдромами в анализаторе синдрома. Одновременно циклически сдвигается кодовое слово в буферном регистре.

Позиции ошибок обнаруживаются при удовлетворении какого-либо неравенства в анализаторе синдрома; на выходе соответствующей схемы анализатора появляется сигнал, по которому выход синдромного регистра подключается ($п_2$=1) к сумматору в цепи циклического сдвига буферного регистра для исправления ошибок.

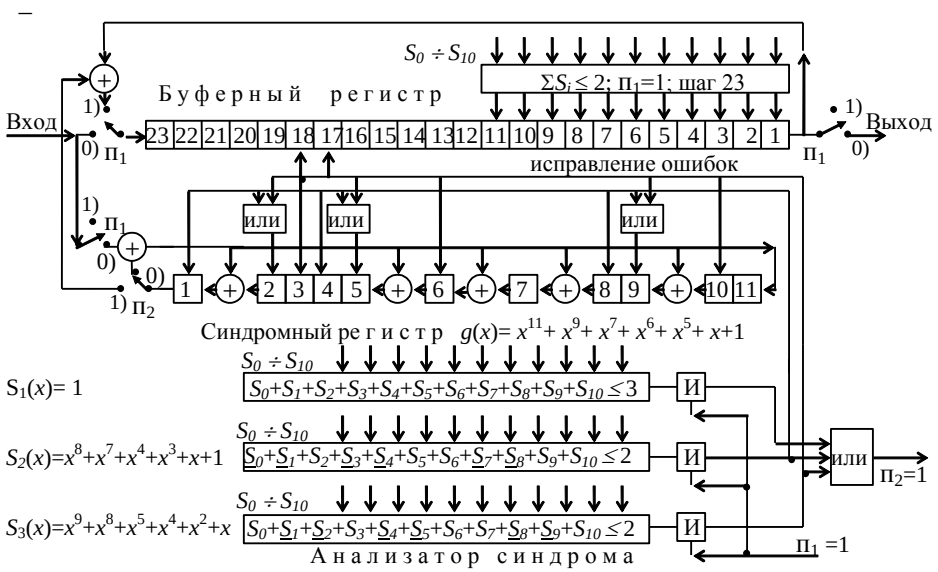



Рис. 2.2. Декодер Касами – Рудольфа циклического кода (23, 12)

Если срабатывает вторая или третья схемы анализатора, то дополнительно исправляются ошибки в 17-й или 18-й ячейках буферного регистра в соответствии с номером покрывающего синдрома; одновременно производится стирание этого синдрома в синдромном регистре. После 23-го цикла производится проверка состояния синдромного регистра и, если остаток не превышает двух единиц, его содержимое используется для коррекции состояний первых 11 ячеек буферного регистра.

На этом декодирование заканчивается и на выход выдаются информационные символы, расположенные в первых 11 ячейках буферного регистра; одновременно на вход может подаваться новое кодовое слово ($п_1=0$).

## 2.3. Мажоритарное и пороговое декодирование циклических кодов

К числу методов, использующих для декодирования алгебраические особенности линейных кодов, относится мажоритарный метод декодирования циклических кодов. Этот метод основан на возможности составления *системы проверок* для каждого символа принятого сообщения. Возможность составления такой системы проверок вытекает из соотношения (1.35).

Каждая $i$-я строка проверочной матрицы **H** задаёт одно соотношение, связывающее между собой символы кодового слова ($a_0$, $a_1$, $a_2$, $a_3$, …, $a_n$):

$$h_{i0}\,a_0 + h_{i1}\,a_1 + \ldots + h_{i\,n\text{-}\setminus}\,a_{n\text{-}1} = 0,\ \ i = 1, 2,3, \ldots, n - k.$$

Набор этих соотношений называется контрольными проверками, на основе которых формируется *система разделённых проверок,* удовлетворяющих следующим требованиям:

1) проверяемый символ $a_i$ входит в каждую контрольную проверку;

2) любой другой символ $a_j$ ($j \neq i$) входит лишь в одну проверку.

Благодаря этому в системе разделённых проверок каждая ошибка в проверяемом символе $a_i$ искажает все проверки, а ошибка в символе $a_j$ ($j \neq i$) искажает только одну проверку, что позволяет принимать решение о значении символа $a_i$ по большинству проверок. Очевидно, что для исправления $t$ ошибочных символов, необходимо иметь $(2\,t+1)$ проверочных соотношений.

В группе циклических кодов можно найти подгруппу, позволяющую построить систему разделённых проверок. Причём для циклических кодов достаточно задать систему проверок одного из символов кодового слова. Эта же система проверок используется для исправления остальных символов после соответствующей циклической перестановки принятого кодового слова.

На рис. 2.3 приведена структурная схема мажоритарного декодера для циклического кода (7,3). Код задается порождающим многочленом $g(x)=x^4+x^3+x^2+1$, а его проверочная матрица **H** и система разделённых проверок имеют вид

$$\mathbf{H} = \begin{vmatrix} 1\,1\,0\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 1\,1\,1\,0\,0\,1\,0 \\ 1\,0\,1\,0\,0\,0\,1 \end{vmatrix}, \qquad \begin{aligned} h_1 &= 1\,1\,0\,1\,0\,0\,0\,, \\ h_2 &= 1\,0\,0\,0\,1\,1\,0\,, \\ h_3 &= 1\,0\,1\,0\,0\,0\,1\,. \end{aligned} \qquad (2.12)$$

Второе проверочное соотношение образовано суммированием по модулю 2 2-й и 3-й строк проверочной матрицы. Дополняя систему тривиальным соотношением $a_1 = a_1$, получаем разделённые проверки для символа $a_1$ в виде

$$
\begin{aligned}
a_1 &= a_1, \\
a_1 &= a_2 \oplus a_4, \\
a_1 &= a_5 \oplus a_6, \\
a_1 &= a_3 \oplus a_7.
\end{aligned}
\qquad (2.13)
$$

Декодирующее устройство состоит из буферного регистра и мажоритарного элемента (М). Принятое кодовое слово записывается в буферный регистр. Решение о значении принятого символа принимается по большинству проверок, исправленный первый символ считывается с выхода мажоритарного элемента, поступает на выход декодера и через ключ – в седьмую ячейку регистра. В результате циклических сдвигов принятого кодового слова в буферном регистре исправляются остальные символы кодового слова. Таким образом, декодирование происходит в течение 14 тактов: первые 7 тактов – ввод принятого кодового слова в буферный регистр (ключ в нижнем положении), следующие 7 тактов – обнаруживаются и исправляются ошибки в символах принятого слова.

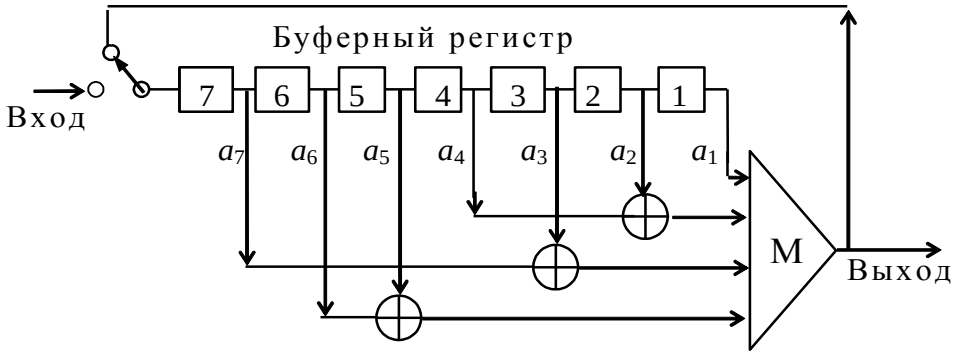



Рис. 2.3. Мажоритарный декодер циклического кода

Мажоритарный метод декодирования может быть реализован также в виде порогового декодера (рис. 2.4). Пороговый декодер является синдромным декодером, у которого анализатор синдрома определяет значение символа ошибки (0 или 1) в декодируемом кодовом символе мажоритарным методом (по большинству элементов синдрома $S_i$ ). Элементы синдрома ошибки ($S_0$ ,$S_1$ ,$S_2$ ,$S_3$) определяются в регистре синдрома путём деления принятого кодового слова на производящий многочлен (ключ К замкнут). В течение следующих семи тактов ведётся поиск и исправление ошибки (ключ К разомкнут). Сложение по модулю 2 первого и третьего элементов синдрома обеспечивает ортогональность проверок.

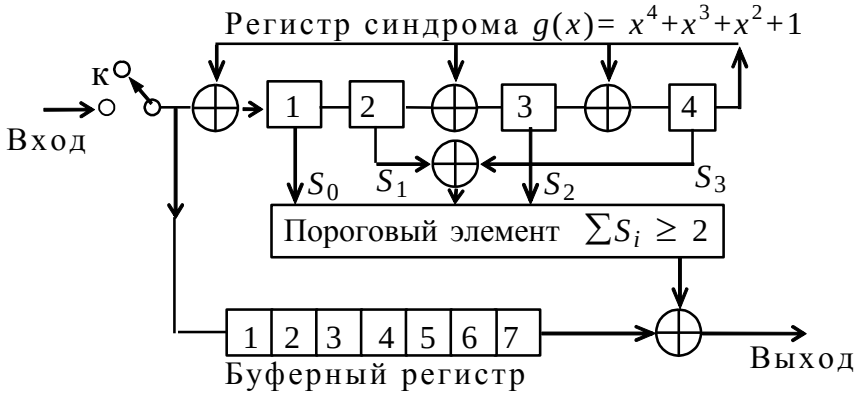



Рис. 2.4. Пороговый декодер циклического кода (7, 3)

В пороговом элементе сумма элементов синдрома на каждом такте сдвига регистра синдрома и буферного регистра сравнивается с порогом, который для данного кода равен 2. Пороговый элемент выдаёт на выходе символ 1, когда сумма элементов синдрома ≥ 2; при этом очередной символ, поступающий с выхода буферного регистра, инвертируется в сумматоре по модулю 2 (происходит исправление обнаруженной ошибки).

Мажоритарный метод является наиболее простым из известных методов декодирования. Однако он применим лишь для небольшой группы линейных кодов (кодов, получающихся из последовательностей максимальной длины, кодов Рида – Маллера и некоторых других циклических и свёрточных кодов [2] ).

## 2.4. Универсальный синдромно-матричный кодек циклических кодов

В качестве примера на рис. 2.5 и 2.6 приведены структурные схемы кодера и декодера циклического кода (15,5), имеющего кодовое расстояние $d$ = 7 и исправляющего любые конфигурации ошибок кратности 3. Производящий многочлен кода $g(x)$ показан на рисунках.

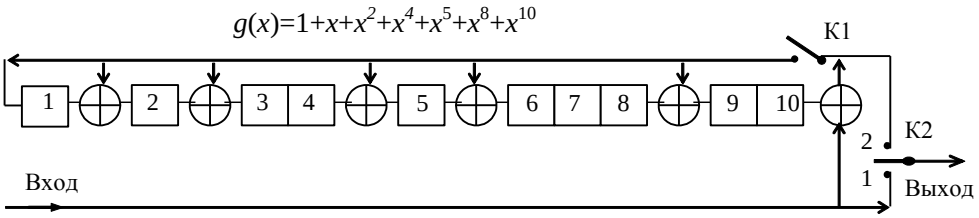



Рис. 2.5. Кодер циклического кода (15, 5)

Кодер представляет собой двоичный регистр сдвига с обратной связью через сумматоры по модулю 2, которые расположены в позициях, определяемых видом производящего многочлена. Кодирование производится за 15 тактов. В течение первых 5 тактов ключ К1 замкнут, ключ К2 находится в положении 1; происходит вычисление проверочных символов в регистре сдвига путем деления поступающих на вход информационных символов на производящий многочлен, одновременно информационные символы через ключ К2 подаются на выход кодера. В течение последующих 10 тактов ключ К1 разомкнут, ключ К2 находится в положении 2; происходит считывание на выход кодера проверочных символов из ячеек регистра сдвига.

Декодер рассматриваемого кода (см. рис. 2.6) построен по синдромно-матричной схеме.


Рис. 2.6. Декодер циклического кода (15, 5)

Вначале при включении декодера производится вычисление матрицы синдромов одиночных ошибок (2.14), таких синдромов у данного кода 15 (при необходимости экономии объёма памяти для хранения синдромов кодов большой длины могут вычисляться и храниться только синдромы ошибок в информационных символах $S_1(x) \div S_k(x)$, поскольку вид синдромов ошибок в проверочных символах $S_{k+1}(x) \div S_n(x)$ очевиден и одинаков для всех кодов).

$$\mathbf{S}^1(x) = \begin{bmatrix} 1010011011 \\ 1111010110 \\ 0111101011 \\ 1001101110 \\ 0100110111 \\ 1000000000 \\ 0100000000 \\ 0010000000 \\ 0001000000 \\ 0000100000 \\ 0000010000 \\ 0000001000 \\ 0000000100 \\ 0000000010 \\ 0000000001 \end{bmatrix} \begin{matrix} S_1(x) \\ S_2(x) \\ S_3(x) \\ S_4(x) \\ S_5(x) \\ S_6(x) \\ S_7(x) \\ S_8(x) \\ S_9(x) \\ S_{10}(x) \\ S_{11}(x) \\ S_{12}(x) \\ S_{13}(x) \\ S_{14}(x) \\ S_{15}(x) \end{matrix} \qquad (2.14)$$

Дополнительно в декодере хранится нулевой синдром $S_0(x)$ = = 0000000000, указывающий на отсутствие ошибок.

Затем принятое кодовое слово поступает в буферный регистр (для хранения на время декодирования) и в регистр синдрома для вычисления синдрома ошибки этого слова $S(x)$ (как и в кодере, в регистре синдрома производится деление принятого кодового слова на производящий многочлен, элементы синдрома $S(x)$: $S_0$, $S_1$, $S_2$, $S_3$, $S_4$, $S_5$, $S_6$, $S_7$, $S_8$, $S_9$ представляют собой остаток от деления). В схеме сравнения 1 синдромы $S_0(x) \div S_n(x)$ сравниваются с синдромом ошибки декодируемого кодового слова.

Если $S(x) = S_0(x)$, то ошибок нет, на выходе схемы сравнения 1 сигнал “да” и декодирование сводится к считыванию информационных символов из буферного регистра на выход декодера.

Если $S(x) \neq S_0(x)$, то начинается операция поиска конфигурации одиночной ошибки; для этого синдромы одиночных ошибок поочерёдно подаются на схему сравнения 1. При обнаружении равенства $S(x) = S_i(x)$, $i = 1 \ldots 15$ на выходе схемы сравнения 1 появляется сигнал «да», по которому в блоке формирования ошибок декодера формируется последовательность ошибок с символом 1 в позиции $i$.

Если $S(x) \neq S_i(x)$, то начинается операция поиска конфигурации двойной ошибки; для этого в блоке вычисления синдромов двойных

ошибок производится попарное сложение элементов строк матрицы синдромов (2.14) и сравнение суммы $S_{ij}(x)$ с синдромом $S(x)$ в схеме сравнения 2. При обнаружении равенства $S(x) = S_{ij}(x)$, $i = 1 \ldots n$, $j = 1 \ldots (n\text{-}1)$ на выходе схемы сравнения 2 появляется сигнал «да», по которому формируется последовательность ошибок с символами 1 в позициях $i$ и $j$.

Если $S(x) \neq S_{ij}(x)$, то начинается операция поиска конфигурации тройной ошибки; для этого в блоке вычисления синдромов тройных ошибок производится поочерёдное сложение элементов трёх строк матрицы синдромов (2.20) и сравнение суммы $S_{ijk}(x)$ с синдромом $S(x)$ в схеме сравнения 3 до обнаружения равенства $S(x) = S_{ijk}(x)$, $i = 1 \ldots n$, $j = 1 \ldots (n\text{-}1)$, $k = 1 \ldots (n\text{-}2)$.

В сумматоре по модулю 2 на выходе буферного регистра происходит сложение символов декодируемого кодового слова и символов последовательности ошибок $e_{ijk}(x)$ и на выход декодера подаётся декодированная последовательность информационных символов.

Как известно, существенным недостатком синдромно-матричного декодера является малое быстродействие при исправлении ошибок большой кратности из-за быстрого увеличения числа синдромов с ростом кратности исправляемых ошибок. В рассматриваемом примере код имеет $n = 15$ синдромов одиночных ошибок, $n\times(n\text{-}1)=15\times14=210$ синдромов двойных ошибок и $n\times(n\text{-}1)\times(n\text{-}2)= =15\times14\times13=2730$ синдромов тройных ошибок.

Для более длинных кодов число возможных синдромов ошибок может быть необозримо велико. При необходимости кратность исправляемых данным кодом ошибок может быть ограничена. Для этого сокращается число установленных в схеме декодера блоков вычисления синдромов, а по сигналу перехода управления к этому блоку формируется сигнал «отказ от декодирования», по которому декодирование прекращается, а по дополнительному выходу сообщается об обнаружении ошибки неисправляемой кратности.

# 3. СВЁРТОЧНЫЕ КОДЫ

## 3.1. Свёрточные коды и их свойства

Свёрточными кодами являются древовидные коды [2], на которые накладываются дополнительные ограничения по линейности и постоянстве во времени. Так же, как и для циклических (блочных) кодов, для сверточных кодов справедлива линейная свертка

$$c_i = \sum_{k=0}^{n-1} g_{i-k} \cdot d_k, \qquad (3.1)$$

или в виде многочленов $c(x) = g(x)\, d(x)$, где $c_i$ – символы кода; $g_{i-k}$ – весовые коэффициенты (коэффициенты производящего многочлена кода $g(x)$); $d_k$ – информационные символы кода.

В отличие от блочных кодов сверточный $(n,k)$ код обычно задаётся скоростью $R=k_0 / n_0$ и требует для своего описания несколько порождающих (производящих) многочленов, которые могут быть объединены в матрицу

$$\mathbf{G}(x) = [\, g_{ij}(x) \,] = \begin{vmatrix} g_{11}(x) & g_{12}(x) & \dots & g_{1n_0}(x) \\ g_{21}(x) & g_{22}(x) & \dots & g_{2n_0}(x) \\ \dots & \dots & \dots & \dots \\ g_{k_0 1}(x) & g_{k_0 2}(x) & \dots & g_{k_0 n_0}(x) \end{vmatrix}, \qquad (3.2)$$

где $i = 1...k_0$, $j = 1...n_0$, $k_0$ и $n_0$ – целые числа: $n_0$ – минимально возможная длина кодового слова при данной скорости $R$; $k_0$ – число информационных символов в этом слове (иногда такое кодовое слово называют «кадром», так как свёрточные коды являются непрерывными и деление последовательности символов кода на кодовые слова может быть только условным).

Скорость кода $R=k_0 / n_0$ определяет избыточность кода и равна отношению скоростей на входе и выходе кодирующего устройства (кодера).

Длина кодового слова $n$ при данной скорости $R$ определяется не только $n_0$, но также длиной (или максимальной степенью) производящих многочленов

$$n = n_0 \max_{i,j} \lfloor \deg g_{ij}(x) + 1 \rfloor, \qquad (3.3)$$

при этом информационная длина кодового слова

$$k = k_0 \max_{i,j} \lfloor \deg g_{ij}(x) + 1 \rfloor.$$

Вместо длины кодового слова часто используется понятие *длины кодового ограничения* $n_a$, которая показывает максимальное расстояние между позициями информационных символов, участвующих в формировании проверочного символа данного кода (например, при $R$ =1/2 длина кодового ограничения равна числу ячеек памяти регистра сдвига кодера)

$$n_a = \sum_{i=1}^{k0} \max [\deg g_{ij}(x) + 1]. \qquad (3.4)$$

Существуют и другие определения длины кодового ограничения, например в [2, 8].

Входная последовательность из $k$ информационных символов представляется вектором-строкой

$$\mathbf{D}(x) = [d_i(x)] = [d_1(x), d_2(x), \ldots d_{k0}(x)],$$

а кодовое слово на выходе кодера $\mathbf{C}(x) = [c_j(x)] = [c_1(x) \ldots c_{n0}(x)]$.

Операция кодирования представляется в виде произведения

$$\mathbf{C}(x) = \mathbf{D}(x)\mathbf{G}(x),$$

или

$$c_j(x) = \sum_{i=1}^{k0} d_i(x)\, g_{ij}(x).$$

Проверочная матрица H($x$) должна удовлетворять условию

$$\mathbf{G}(x) \cdot \mathrm{H}^{\mathrm{T}}(x) = 0, \qquad (3.5)$$

а вектор синдромов ошибки (синдромных многочленов) равен

$$\mathbf{S}(x) = \mathbf{V}(x) \cdot \mathrm{H}^{\mathrm{T}}(x) = [S_j(x)] = [S_1(x) \ldots S_{n0-k0}(x)],$$

то есть $(n_0 - k_0)$-мерный вектор-строка из многочленов.

$$\mathbf{V}(x) = \mathbf{C}(x) + \mathbf{e}(x),$$

где $\mathbf{e}(x)$ – вектор ошибок в декодируемой последовательности $\mathbf{V}(x)$.

Очевидно, что $\mathbf{S}(x) = \mathbf{V}(x)\cdot \mathrm{H}^{\mathrm{T}}(x) = \mathbf{e}(x)\cdot \mathrm{H}^{\mathrm{T}}(x)$. (3.6)

В дальнейшем будут рассматриваться только двоичные свёрточные коды, для которых сложение и умножение многочленов производится над двоичным полем GF(2).

Если свёрточный код является систематическим, то информационные символы на выходе кодера совпадают с информационной последовательностью на входе кодера, то есть $g_1(x) = 1$; тогда

$$\mathbf{G}(x) = [\ 1,\ g_2(x),\ g_3(x)\ \ldots\ g_{n0}(x)], \qquad (3.7)$$

а для кодов $R = (k_0/n_0) = 1/2$, $\mathbf{G}(x) = [1,\ g(x)]$.

Систематические сверточные коды всегда являются некатастрофическими, то есть всегда удовлетворяют условию

$$\text{НОД}[\ g_1(x)\ \ldots\ g_{n0}(x)] = \ x^{\alpha} = 1$$

в отличие от катастрофических, имеющих склонность к неограниченному размножению некоторых конфигураций ошибок в процессе декодирования.

Такие коды могут декодироваться с помощью алгоритма для многочленов, согласно которому всегда существуют многочлены $a_1(x)\ \ldots\ a_{n0}(x)$ такие, что $a_1(x)\cdot g_1(x) + \ldots + a_{n0}(x)\cdot g_{n0}(x) = 1$.

При этом процесс кодирования осуществляется в виде

$$c_j(x) = d(x)\cdot\ g_j(x), \qquad i = 1\ldots n_0,$$

а декодирования

$$d(x) = a_1(x)\cdot c_1(x) + \ldots + a_{n0}(x)\cdot c_{n0}(x).$$

Наибольший общий делитель (НОД) двух многочленов $S(x)$ и $r(x)$ над полем GF(q) можно вычислить с помощью итеративного применения деления многочленов: если $\deg\ [S(x)] \geq \deg\ [r(x)] \geq 0$, то алгоритм вычисления НОД имеет вид

$$S(x) = a_1(x)\cdot r(x) + r_1(x),$$

$$r(x) = a_2(x)\cdot r_1(x) + r_2(x),$$

$$r_1(x) = a_3(x)\cdot r_2(x) + r_3(x),$$

$$\ldots\ldots\ldots\ldots\ldots$$

$$r_{n-1}(x) = a_{n+1}(x)\cdot r_n(x)\ , \qquad (3.8)$$

процесс прекращается при получении нулевого остатка $r_i(x)$: тогда последний ненулевой остаток

$$r_n(x) = \text{НОД}\,[r(x), S(x)].$$

Сверточные коды, скорость которых $R \neq 0{,}5$, позволяют либо увеличить скорость кода $R = (n_0 - 1)/n_0$ за счет потери корректирующей способности, либо увеличить корректирующую способность кода за счет уменьшения скорости $R = 1/n_0$. Как уже отмечалось, свёрточные коды для практического применения не должны позволять катастрофического размножения ошибок. Поэтому для кодов $R = 1/n_0$ порождающие многочлены не должны иметь общего множителя; а для кодов $R = (n_0 - 1)/n_0$ не должны иметь общего множителя определители различных $|k_0 \times n_0|$ подматриц. При этом обеспечивается ортогональность всех проверочных соотношений набора порождающих многочленов; практически целесообразно также иметь в каждом из многочленов одинаковое число ненулевых членов.

Качество работы декодера зависит также от длины блока декодирования – $L$ (ширины окна декодирования), причём чем больше блок декодирования, тем лучше. Это определяется тем, что сверточный код имеет множество минимальных расстояний, которые определяются длинами начальных сегментов кодовых блоков, между которыми определяется минимальное расстояние. Это расстояние равно $d_L$ – числу ненулевых символов в блоке декодирования $L$ с ненулевым первым кадром. При этом декодер может исправить $t$ ошибок, если $2t + 1 \leq d_L$. Как будет показано в п. 3.3, для большинства хороших сверточных кодов, декодируемых с обратной связью и итерациями, длина блока декодирования может быть равна $n$.

*Свободным кодовым расстоянием* сверточного кода называется $d = \max d_L$ , которое равно наименьшему весу ненулевого начального блока декодирования при $L \to \infty$. Как правило, свободное кодовое расстояние $d \le (n - k)$, но в общем случае может быть и несколько больше. В табл. 3.1 приведены некоторые несистематические сверточные коды с максимальным свободным расстоянием [2].

Табл. 3.1. *Несистематические свёрточные коды с большим d*

| Параметры кода | | | | Порождающие многочлены | | | |
|---|---|---|---|---|---|---|---|
| $R$ | $n$ | $k$ | $D$ | $g_1$ | $g_2$ | $g_3$ | $g_4$ |
| 1/2 | 10 | 5 | 7 | 11001 | 10111 | – | – |
| 1/3 | 15 | 5 | 12 | 10101 | 11011 | 11111 | – |
| 1/4 | 20 | 5 | 16 | 10101 | 11011 | 10111 | 11111 |

*Оптимальными* будем считать такие коды, которые обеспечивают в заданном канале меньшую вероятность ошибок декодирования $P_д$ при одинаковой скорости передачи информации $R = k_0 / n_0$ и одинаковой вычислительной сложности декодирования.

Если при поиске хороших блочных кодов широко используются алгебраические методы, то для отбора близких к оптимальным сверточных кодов обычно используются различные алгоритмы просмотра большого числа кодов на ЭВМ. Можно сказать, что регулярных методов поиска сверточных кодов с большим кодовым ограничением не существует. Большинство известных в настоящее время хороших сверточных кодов найдено поиском на ЭВМ [2].

Сверточные коды в ряде практически важных случаев позволяют получить энергетически лучшие результаты в сравнении с блочным. В [2, 3, 8] приводятся результаты исследования эффективности некоторых сверточных кодов. У сверточных кодов при уменьшении скорости кода значительно медленнее обмен эффективности на удельную скорость. Современная элементная база позволяет реализовать близкие к оптимальным алгоритмы декодирования сверточ-

ных кодов с большим кодовым ограничением, что позволяет увеличить энергетический выигрыш. В сочетании с итерационными и многоступенчатыми пороговыми алгоритмами декодирования эти коды позволяют приблизиться к предельной эффективности для двоичных кодов как при жестких, так и мягких решениях в канале [3, 8, 9].

Кроме рассмотренного выше применяются и другие способы описания свёрточных кодов с помощью:

– кодового дерева или решетчатой структуры;

– разностных треугольников.

Эти способы описания обычно используются для построения различных алгоритмов декодирования свёрточных кодов и будут рассмотрены ниже.

## 3.2. Кодирование и декодирование свёрточных кодов

### 3.2.1. *Методы кодирования и декодирования*

Кодирование свёрточных кодов производится аналогично кодированию блочных циклических кодов с помощью регистров сдвига, у которых структура обратных связей определяется производящим многочленом кода. Различие только в том, что при $k_0 > 1$ свёрточный код имеет несколько производящих многочленов, а кодер должен иметь соответствующее число регистров сдвига.

На рис. 3.1 приведены функциональные схемы двух кодеров для свёрточных кодов $R=1/2$ и $R=2/3$ [8].

Коды являются несистематическими; производящие многочлены кода $R=1/2$ представлены на рис. 3.1. Кодеры работают следующим образом. На вход регистра сдвига кодера (рис. 3.1, *а*) из 3 ячеек памяти подаётся двоичная последовательность информационных символов, из которых с помощью регистра сдвига и сумматоров по модулю 2 формируются две двоичные последовательности.

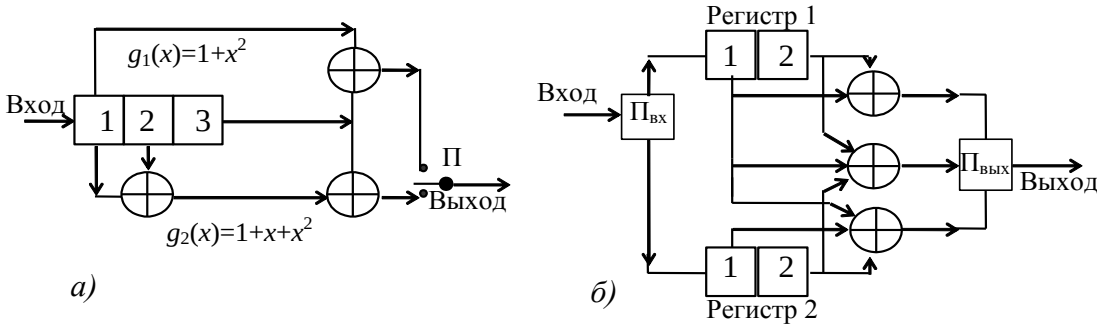



Рис. 3.1. Кодеры сверточных кодов: *а) R*=1/2 и *б) R*=2/3

Символы этих последовательностей с помощью переключателя П поочерёдно подключаются к выходу кодера; скорость переключения должна быть в два раза больше скорости ввода информационных символов.

Матрица производящих многочленов кода *R*=2/3 (рис. 3.1, *б*) имеет вид

$$\mathrm{G}(x)=\begin{bmatrix} g_{11}(x), g_{12}(x), g_{13}(x) \\ g_{21}(x), g_{22}(x), g_{23}(x) \end{bmatrix}=\begin{bmatrix} 1+x, & 1+x, & 1 \\ 0, & x, & 1+x \end{bmatrix}. \qquad (3.9)$$

Регистры сдвига 1 и 2 (число регистров равно $k_0$) имеют по две ячейки памяти и три сумматора по модулю 2 (число сумматоров равно $n_0$), формирующих символы кода в соответствии с видом производящих многочленов. Переключатель $П_{вх}$ разделяет входные информационные символы между регистрами, переключатель $П_{вых}$ формирует кодовую последовательность на выходе кодера из выходных символов сумматоров.

Кодовое дерево рассматриваемого кода *R*=1/2 и соответствующая ему кодовая решётка имеют вид, показанный на рис. 3.2.

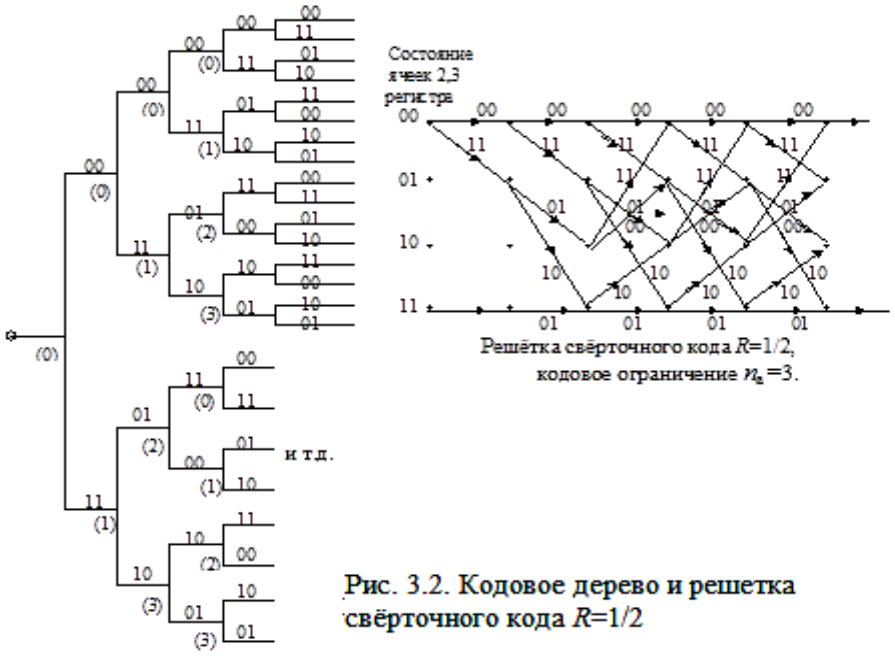
Кодовое дерево строится таким образом, что информационному символу «0» соответствует перемещение на верхнюю ветвь (ребро) дерева, а информационному символу «1» – на нижнюю ветвь. Можно обратить внимание, что после формирования четырех вершин (на рисунке отмечены цифрами 0, 1, 2, 3 в скобках) структура ветвей дерева повторяется. Это обстоятельство определяется состоянием двух последних ячеек памяти регистра сдвига кодера (00, 01, 10, 11); в общем

случае число состояний зависит от кодового ограничения кода и равно $2^{n_a-1}$. Решётка сверточного кода представляет состояния кодера в виде четырёх уровней, а ветви дерева являются рёбрами решётки, в результате чего избыточные части кодового дерева отождествляются. Такое представление кода является более удобным при разработке и описании процессов декодирования.

Если свёрточный код является систематическим, то $g_1(x) = 1$ (в кодере рис. 3.1, *а* отсутствует верхний сумматор) и информационная последовательность становится частью выходной последовательности без кодирования, что упрощает в последующем процесс её декодирования.

Для декодирования свёрточных кодов используются в основном три метода [8]: декодирование по алгоритму Витерби, синдромное декодирование и последовательное декодирование.

Декодирование по алгоритму Витерби и последовательное декодирование являются оптимальными по критерию максимального правдоподобия и довольно широко используются для двоичных сверточных кодов с малой длиной кодового ограничения в каналах с жестким и мягким решениями.




Решётка свёрточного кода $R$=1/2, кодовое ограничение $n_a$=3.

Рис. 3.2. Кодовое дерево и решетка свёрточного кода $R$=1/2

Синдромные методы декодирования являются субоптимальными и основаны на вычислении и последующем анализе синдрома ошибок, что делает их похожими на соответствующие методы декодирования блочных (например, циклических) кодов: это **декодирование с табличным поиском и пороговое декодирование**.

Известно, что пороговое декодирование возможно для относительно ограниченного класса сверточных кодов [11], позволяющих построить ортогональные проверки относительно контролируемого (проверяемого) символа $e_0^{u}$. В настоящее время получено множество хороших кодов, состоящее из канонических самоортогональных кодов (КСОК) и кодов, допускающих ортогонализацию, в том числе сверточных кодов, полученных на ЭВМ методом проб и ошибок. Списки таких кодов приведены в работах [8, 11, 12, 13].

Для построения и описания ортогональных свёрточных кодов широко используется метод **разностного треугольника** [8]. Поясним этот метод на примере, учитывая, что большинство таблиц самоортогональных и ортогонализируемых кодов составлены в виде разностных треугольников.

Разностный треугольник представляет собой треугольный массив, вычисленный по первой строке проверочной матрицы $H^T$ так, что $ij$-й элемент треугольника равен разности между номерами столбцов, в которых $(i+j)$-й и $j$-й символы этой строки равны 1.

Рассмотрим разностный треугольник кода (406,203), $R$=1/2, который в дальнейшем часто используется в практических приложениях.

$$\Delta_p = \begin{array}{ccccccccc} 7 & 20 & 49 & 37 & 24 & 18 & 1 & 14 & 32 \\ 27 & 69 & 86 & 61 & 42 & 19 & 15 & 46 & \\ 76 & 106 & 110 & 79 & 43 & 33 & 47 & & \\ 113 & 130 & 128 & 80 & 57 & 65 & & & \\ 137 & 148 & 129 & 94 & 89 & & & & \\ 155 & 149 & 143 & 126 & & & & & \\ 156 & 163 & 175 & & & & & & \\ 170 & 195 & & & & & & & \\ 202 & & & & & & & & \end{array} \qquad (3.10)$$

Заметим, что в этом треугольнике нет одинаковых чисел, а первый столбец представляет собой запись степеней производящего многочлена этого кода

$$g(x) = 1 + x^7 + x^{27} + x^{76} + x^{113} + x^{137} + x^{155} + x^{156} + x^{170} + x^{202}; \quad (3.11)$$

с другой стороны,

$$g(x) = 1 + x^{\Delta_{11}} + x^{\Delta_{11}+\Delta_{12}} + \ldots + x^m, \quad (3.12)$$

где $m = \sum_{j=1}^{J-1} \Delta_{1j}$, $\Delta_{1j}$ – элемент первой строки разностного треугольника.

Следовательно, для построения кода достаточно знать первую строку разностного треугольника. Первая строка проверочной матрицы $H^T_{202}$ соответствует многочлену $g(x)$, записанному в двоичной форме; остальные строки образуются сдвигами первой строки $g(x) \cdot x^i$, $i = 1...m$ (при этом степени $g_i(x)$ суммируются по mod $m$, то есть элементы матрицы слева от диагонали заменяются нулями). Тогда множество ортогональных проверок (элементов синдрома ошибки) относительно проверяемого символа ошибки $e_k^u$ (в $k$-м информационном символе) записывается следующим образом:

$$\begin{aligned}
S_k &= e_k^u + e^u_{k-7} + e^u_{k-27} + \ldots e^u_{k-202} + e_k^p, \\
S_{k+7} &= e^u_{k+7} + e_k^u + e^u_{k-20} + e^u_{k-69} + \ldots e^u_{k-195} + e^p_{k+7}, \\
S_{k+27} &= e^u_{k+20} + e^u_{k+27} + e_k^u + e^u_{k-49} + e^u_{k-86} + \ldots e^u_{k-175} + e^p_{k+27}, \\
S_{k+76} &= e^u_{k+49} + e^u_{k+69} + e^u_{k+76} + e_k^u + e^u_{k-37} + e^u_{k-71} + \ldots e^u_{k-136} + e^p_{k+76}, \\
S_{k+113} &= e^u_{k+37} + \ldots e_k^u + e^u_{k-24} + \ldots e^u_{k-89} + e^p_{k+113}, \\
S_{k+137} &= e^u_{k+24} + e^u_{k+71} + \ldots e_k^u + e^u_{k-18} + \ldots e^u_{k-65} + e^p_{k+137}, \\
S_{k+155} &= e^u_{k+18} + e^u_{k+42} + \ldots e_k^u + e^u_{k-1} + \ldots e^u_{k-47} + e^p_{k+155}, \\
S_{k+156} &= e^u_{k+1} + e^u_{k+19} + \ldots e_k^u + e^u_{k-14} + e^u_{k-46} + e^p_{k+156}, \\
S_{k+170} &= e^u_{k+14} + \ldots e^u_{k+170} + e_k^u + e^u_{k-32} + e^p_{k+170}, \\
S_{k+202} &= e^u_{k+32} + \ldots e^u_{k+202} + e_k^u + e^p_{k+202}, \quad (3.13)
\end{aligned}$$

где $S_{k+j}$ – элементы синдрома ошибки для $k$-го информационного символа; $j$ – степени производящего многочлена $g(x)$; $e^u_{k+j}$ – символ

ошибки в ($k$+$j$)-м информационном символе; $e^{p}_{k+j}$ – символ ошибки в ($k$+$j$)-м проверочном символе.

Множество проверок (3.13) показывает, что рассматриваемый свёрточный код является каноническим самоортогональным, так как никакие два проверочных уравнения не содержат более одного общего символа (в данном случае это символ $e_k^{u}$ ). Код не будет каноническим самоортогональным, если для формирования ортогональных проверочных уравнений необходимо использовать линейные комбинации элементов синдрома (3.13).

Общее число проверочных уравнений (ортогональных проверок) определяется числом ненулевых элементов производящего многочлена. В данном случае $J$ = 10 (кодовое расстояние $d$ = $J$+1), что позволяет мажоритарным методом правильно декодировать $k$-й информационный символ, если число ошибок в символах, участвующих в проверках, не более пяти.

В общем случае, когда $R = k_0/n_0$, набор порождающих многочленов кода должен быть таким, чтобы все элементы разностных треугольников были различны. Например, для систематического ортогонального кода $R$ = 2/3 (или 1/3), с числом проверок $J$ = 6, $\max\limits_{i,j}\left[\deg g_{ij}(x)\right] = 39$ разностные треугольники могут иметь вид

$$\left|\begin{array}{ccccc} 14 & 7 & 2 & 13 & 3 \\ 21 & 9 & 15 & 16 & \\ 30 & 24 & 31 & & \\ 54 & 55 & & & \\ 109 & & & & \end{array}\right| \quad \left|\begin{array}{ccccc} 8 & 19 & 1 & 4 & 6 \\ 27 & 20 & 5 & 10 & \\ 47 & 25 & 15 & & \\ 72 & 40 & & & \\ 112 & & & & \end{array}\right|, \qquad (3.14)$$

а первые строки этих треугольников задают порождающие многочлены

$g_1(x) = 1 + x^{14} + x^{21} + x^{23} + x^{36} + x^{39}$,

$g_2(x) = 1 + x^{8} + x^{27} + x^{28} + x^{32} + x^{38}$.

Причем существует двойственность между кодами $R = 1/n_0$ и $R = (n_0 - 1)/n_0$, что позволяет использовать для построения этих кодов одинаковые порождающие многочлены.

Систематический сверточный код с $R=(n_0-1)/n_0$ получается в виде

$$C_{n_0}(x)=\sum_{i=1}^{n_0-1} g_i(x)\cdot D_i(x) - \text{проверочный символ,}$$

$$C_i(x)=D_i(x),\ i=1,2,...n_0-1 - \text{информационные символы,} \qquad (3.15)$$

где $D_i(x)$ – информационные символы на входе кодирующего устройства; $C_i(x)$ – кодовые символы на выходе кодирующего устройства.

Систематический сверточный код с $R=1/n_0$ может строиться с использованием этих же производящих многочленов, но теперь для одной информационной последовательности вычисляется ($n_0$–1) проверочная последовательность символов $C_1(x)=D_1(x)$

$$C_i(x)=g_i(x)\cdot D_i(x),\quad i=1,2,...n_0-1. \qquad (3.16)$$

$$\mathbf{H}^T=\begin{vmatrix} g_1(x) \\ x^{14}g_1(x) \\ x^{21}g_1(x) \\ x^{23}g_1(x) \\ x^{36}g_1(x) \\ x^{39}g_1(x) \end{vmatrix} \begin{vmatrix} g_2(x) \\ x^{8}g_2(x) \\ x^{27}g_2(x) \\ x^{28}g_2(x) \\ x^{32}g_2(x) \\ x^{38}g_2(x) \end{vmatrix} \qquad (3.17)$$

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

Усеченная проверочная матрица приведенного выше кода для $R=(n-1)/n_0=2/3$ может быть записана в виде порождающего многочлена $g_1(x)$ в двоичной форме, а седьмая строка – в виде порождающего многочлена $g_2(x)$ в двоичной форме (3.17). Число столбцов этой матрицы равно максимальной степени набора порождающих многочленов $\max\limits_i[\deg\ g_i(x)]+1$. Первая строка матрицы содержит единицы в шести столбцах (число ненулевых элементов многочлена $g_1(x)$), что позволяет записать шесть ортогональных проверочных соотношений для вычисления шести элементов синдрома ошибки $\{S_0, S_{14}, S_{21}, S_{23}, S_{36}, S_{39}\}$ для исправления ошибок в первой информационной последовательности; седьмая строка матрицы (3.17) содержит единицы также в шести столбцах и позволяет вычислить шесть элементов синдрома $\{S_0, S_8, S_{27}, S_{28}, S_{32}, S_{38}\}$ для исправления ошибок во второй информационной последовательности.

Кодовое расстояние рассматриваемого кода $d=J+1$, где $J=6$ – количество проверочных соотношений. Однако в сравнении с кодом $R=1/2$ происходит ухудшение качества декодирования из-за увеличения числа информационных символов в проверочных соотношениях.

Для кода $R=1/3$ проверочная матрица принимает вид

$$\mathbf{H}^{\mathrm{T}}=\left|\begin{array}{c|c} g_1(x) & g_2(x) \\ x^{14}g_1(x) & x^{8}g_2(x) \\ x^{21}g_1(x) & x^{27}g_2(x) \\ x^{23}g_1(x) & x^{28}g_2(x) \\ x^{36}g_1(x) & x^{32}g_2(x) \\ x^{39}g_1(x) & x^{38}g_2(x) \end{array}\right| \qquad (3.18)$$

$$\left|\begin{array}{cccccc|cccccc} 1 & & & & & 0 & 1 & & & & & 0 \\ & 1 & & & & & & 1 & & & & \\ & & 1 & & & & & & 1 & & & \\ & & & 1 & & & & & & 1 & & \\ & & & & 1 & & & & & & 1 & \\ 0 & & & & & 1 & 0 & & & & & 1 \end{array}\right|$$

Число столбцов подматрицы $\mathbf{H}^{\mathrm{T}}$ равно 40, что соответствует $(\max_i[\deg\ g_i(x)]+1)$.

Первая строка этой матрицы содержит единицы в 12 столбцах, что позволяет записать 12 ортогональных проверочных соотношений и вычислить 12 элементов синдрома ошибки $\{S_0, S_{14}, S_{21}, S_{23}, S_{36}, S_{39}, S_{40}, S_{48}, S_{67}, S_{68}, S_{72}, S_{78}\}$. Это позволяет исправить ошибку в проверяемом информационном символе при искажении до 6 символов, участвующих в проверках (кодовое расстояние $d = 2J+1$ или $d = J_{\Sigma}+1$).

Следовательно, для кодов $R = 1/n_0$ число проверочных соотношений $J_{\Sigma} = (n_0 - 1)J$.

Подматрицы проверочной матрицы (3.18) в процессе декодирования могут использоваться отдельно для формирования двух синдромов ошибки $S_1(x)$, $S_2(x)$ (и так далее, если $n_0>3$), а для исправления ошибки в этом случае может применяться мажоритарная процедура на уровне обнаружения (или необнаружения) ошибки каждым синдромом.

В прил. 2 приведен список многочленов, которые могут быть использованы в качестве порождающих как для сверточных кодов $R$=1/2, так и для кодов $R\neq1/2$. Необходимый набор порождающих многочленов формируется с учетом рекомендаций, сформулированных выше.

### 3.2.2. *Декодирование по алгоритму Витерби*

Алгоритм декодирования Витерби свёрточного кода $R$=1/2, кодер которого приведён на рис. 3.1, *а*, а кодовое дерево и решётчатая диаграмма – на рис. 3.2, показан на рис. 3.3.

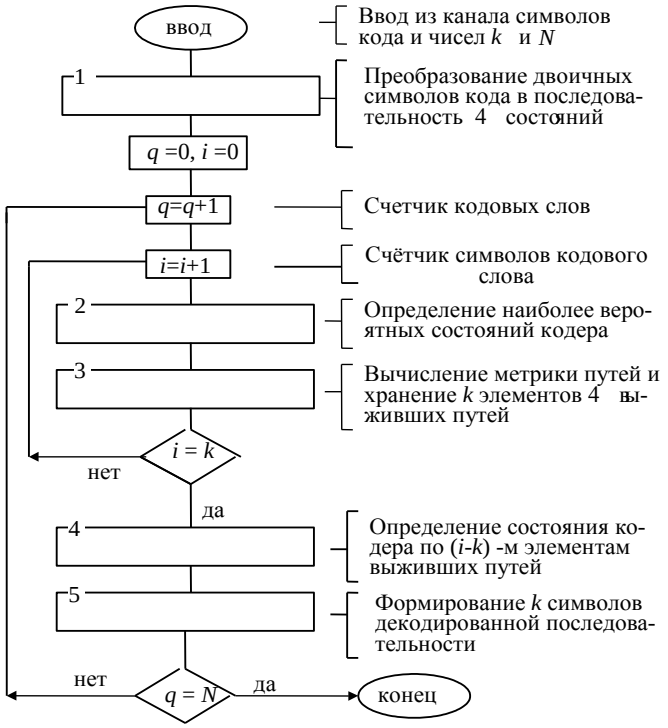



Рис. 3.3. Алгоритм декодирования свёрточного кода $R$=1/2

Декодирование производится словами по $k$ пар двоичных символов канала. В блоке 1 эта последовательность преобразуется в четверичную последовательность (0, 1, 2, 3) по количеству состояний кодера: $2^2$ (основание кода в степени, равной старшей степени производящего многочлена). Декодирование производится по алгоритму Витерби, сущность которого изложена в [8]. Для реализации этого алгоритма $k$ должно быть существенно больше кодового расстояния.

В блоках 2 и 3 производится оценка состояний кодера, вычисление и сохранение относительно этих состояний, метрики четырёх путей изменения состояний с учётом символов канала; при этом в соответствии с метрикой по мере увеличения $i$ всегда сохраняются только 4 (из 8 возможных) выживших пути (ближайшие к декодируемой последовательности). Далее после обработки всех $k$ символов канала первого кодового слова начинается обработка символов второго кодового слова и одновременно в блоке 4 по выжившим путям вычисляется окончательная оценка состояний кодера и определяются все $k$ информационных символов предыдущего кодового слова.

В блоке 5 формируется двоичная последовательность декодированных информационных символов в $N$ информационных слов для статистической обработки.

Верхняя граница вероятности ошибки декодирования при использовании алгоритма Витерби определяется выражением

$$P_{\partial} < \frac{\left\{2\left[p(1-p)\right]^{1/2}\right\}^{5}}{\left\{1-4\left[p(1-p)\right]^{1/2}\right\}^{2}}, \qquad (3.19)$$

где $p$ – вероятность ошибки в канале с независимыми ошибками.

Основные трудности при реализации алгоритма Витерби определяются тем, что сложность декодера экспоненциально растет с увеличением кодового ограничения (число состояний декодера равно $2^{n-1}$); поэтому значение кодового ограничения кодов, применяемых на практике, не превышает $n_a \leq 10$. Недвоичные коды декодировать алгоритмом Витерби еще сложнее.

### 3.2.3 *Последовательное декодирование*

В отличие от алгоритма Витерби при последовательном декодировании производятся продолжение и обновление метрики только одного пути, который представляется наиболее вероятным, при этом делается попытка принять решение о декодируемом символе, принятом в начале пути. Если решение принять невозможно (в соответствии с заданным правилом), производится либо движение вперёд, то есть прием очередного символа и обработка метрики данного пути, или возврат назад, если движение вперёд по данному пути ошибочно (значения метрики увеличиваются). Декодер двигается вперёд и назад, пробуя различные пути, до тех пор, пока не будет принято решение о декодировании символа, расположенного в начале пути.

Основное достоинство последовательного алгоритма заключается в том, что в среднем длина пути, достаточная для правильного декодирования, меньше, чем у алгоритма Витерби. Недостатки определяются тем, что длина пути, приводящего к правильному декодированию, является случайной величиной.

Это вызывает затруднения при реализации декодера, так как нельзя определить заранее, какой объём памяти потребуется для сохранения метрики рассматриваемого пути, а это значит, что всегда существует вероятность переполнения памяти и сбоя декодера.

В практике построения последовательных декодеров применяются два варианта алгоритма последовательного декодирования, позволяющих получить приемлемые для реализации сложность декодирования и объём памяти: алгоритм Фано и стек-алгоритм.

Стек-алгоритм более прост для понимания и реализации на микропроцессорах, но требует, как будет показано ниже, большего объёма памяти. Однако, учитывая развитие микросхемотехники, этот недостаток вряд ли является существенным.

Структурная схема стек-алгоритма показана на рис. 3.4, где $L_0$ – значение метрики пути в исходном состоянии. Декодер создаёт стек, состоящий из просмотренных ранее путей (эти пути могут иметь разную длину), и упорядочивает их в соответствии со значением метрики. На каждом шаге продолжается путь, находящийся наверху стека [8], что порождает $2^{k0}$ новых путей со своей метрикой. Затем стек опять упорядочивается с учётом неиспользованных путей предыдущего состояния стека и процедура продолжается до тех пор, пока путь, находящийся наверху стека, не дойдёт до конца декодируемой последовательности.

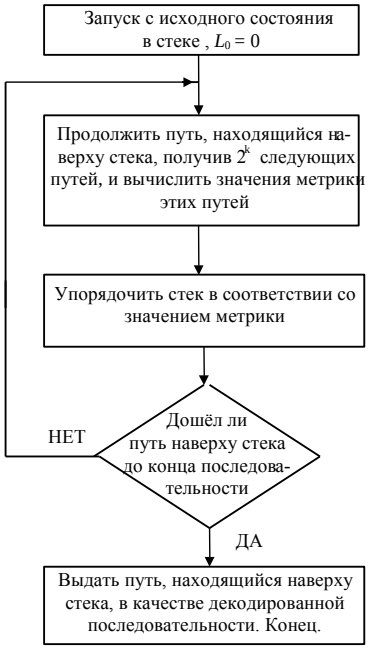



Рис. 3.4. Стек-алгоритм последовательного декодирования

Наиболее сложной операцией в рассматриваемом алгоритме является необходимость упорядочивания стека на каждом шаге декодирования, причём требуемая ёмкость стека случайная величина.

### 3.2.4. *Синдромное декодирование*

Синдромное декодирование свёрточных кодов в принципе не отличается от синдромного декодирования циклических кодов: вначале декодер по принимаемой последовательности символов вычисляет вектор синдромов ошибок (3.6), когда $R \neq 1/2$, или один синдром ошибки, когда $R=1/2$; затем путём анализа синдрома определяется вектор (или символ) ошибки и производится коррекция соответствующих информационных символов входной последовательности.

Практически используются в основном два метода синдромного декодирования:

– с табличным поиском;

– пороговое.

Декодирование с табличным поиском заключается в том, что вычисленный синдром ошибки сравнивается с таблицей всевозможных синдромов ошибок данного кода, для каждого из которых символ ошибки заранее определён. После обнаружения подобного синдрома в таблице остаётся только выполнить коррекцию информационной последовательности.

Очевидно, что декодирование с табличным поиском (как и аналогичное декодирование циклических кодов) практически можно применять только в тех случаях, когда объём таблицы ограничен, иначе декодер получается слишком сложным. Это свёрточные коды с малым кодовым ограничением ($n_a \leq 10 – 20$) и соответственно малым энергетическим выигрышем (1 – 2,5 дБ).

### 3.2.5. *Пороговое декодирование свёрточных кодов*

Наиболее перспективными в смысле благоприятного сочетания качества декодирования и сложности реализации являются пороговые (в частных случаях мажоритарные) алгоритмы декодирования [6, 11] сверточных кодов. В различных модификациях порогового алгоритма декодирования вычислительная сложность слабо зависит от кодового ограничения, а определяется либо кратностью исправляемых ошибок на длине $n$, либо числом проверочных соотношений, используемых для исправления ошибок.

Пороговое декодирование возможно для определенного (хотя и достаточно широкого) класса линейных кодов как блочных, так и сверточных, позволяющих получить так называемые разделенные (или ортогональные) проверки на четность. При этом $J$ проверочных уравнений кода должны удовлетворять определенным требованиям:

– декодируемый (проверяемый) информационный символ должен входить во все проверочные уравнения;

– любой другой символ кода может входить только в одно проверочное уравнение;

– число проверочных уравнений $J$ и число исправляемых ошибок $t$ связаны соотношением $J \geq 2t$ , кодовое расстояние $d_L = J + 1$.

Пороговое декодирование во многих случаях не является оптимальным (в смысле вероятности ошибки декодирования) в сравнении с последовательным декодированием. Однако в практически важных случаях, особенно в каналах с переменными параметрами, эффективность некоторых модификаций пороговых декодеров может быть достаточно высокой.

Алгоритмы порогового декодирования в сочетании с кодами, имеющими малую плотность проверок на четность, являются более гибкими в каналах с пакетирующимися ошибками. Это, например, итерационные алгоритмы [14, 15, 16], алгоритмы многоступенчатого порогового декодирования и другие, которые за счет внутреннего перемежения и итераций позволяют исправлять большие пакеты ошибок. В каналах с известной статистикой шума пороговые декодеры могут эффективно использовать мягкие решения.

Как видно из системы уравнений (3.13), самоортогональный код не требует устранения влияния предыдущих решений декодера для получения системы ортогональных проверок, то есть возможно детерминированное декодирование, и каждый элемент синдрома ошибки образуется в результате суммирования одинакового числа шумовых символов $e_k$. Множество элементов синдрома $\{S_k, S_{k+7}, \ldots, S_{k+202}\}$ задает множество проверок $\{A_l\}$, ортогональное относительно символа ошибки $e^u_k$ в $k$-м информационном символе, которое может быть использовано для порогового (мажоритарного) декодирования.

$$\sum_{l=1}^{J} A_l > J/2 \text{, тогда } e_k^u = 1, \text{ иначе } e_k^u = 0. \qquad (3.20)$$

Структурная схема порогового декодера, реализующего правило решения (3.20), показана на рис. 3.5.

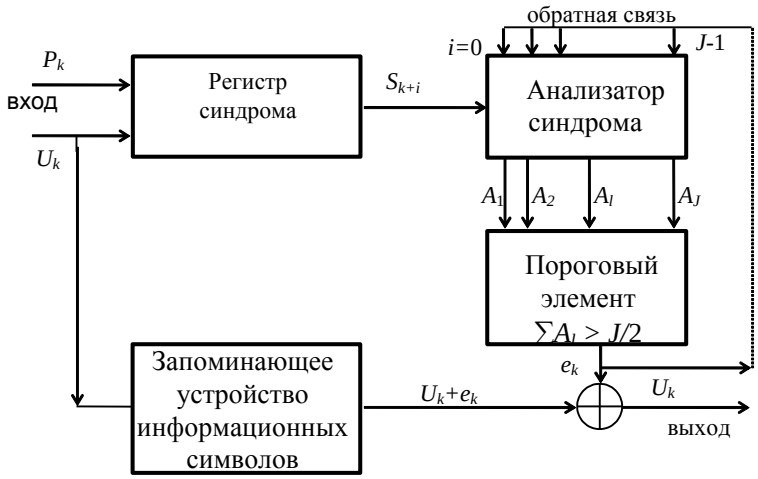



Рис. 3.5. Функциональная схема порогового декодера

Последовательность символов канала после разделения на информационные $U_k$ и проверочные $P_k$ поступает в регистр синдрома, где производится вычисление элементов синдрома $S_k$ . В анализаторе синдрома формируются ортогональные проверки $A_l$ ($l$=1…$J$), сумма которых в пороговом устройстве сравнивается с уровнем порога, и определяется значение символа ошибки $e_k$ . Исправление ошибки происходит в сумматоре по модулю два, на второй вход которого подаются информационные символы канала из запоминающего устройства, выполняющего функцию хранения этих символов на время вычисления символа ошибки.

К достоинствам порогового декодера относится возможность его реализации на скоростях передачи, практически равных скорости работы элементной базы, используемой для построения декодера.

Качество порогового декодера можно улучшить, если ввести обратную связь (на рис. 3.5 показана пунктиром), через которую решение декодера подаётся в анализатор синдрома и устраняет влияние ошибки данного символа на декодирование следующего символа. При декодировании с обратной связью кодовое расстояние $d_L$ больше, чем при детерминированном декодировании (без обратной связи) и ближе к значению свободного кодового расстояния $d$, так как при детерминированном декодировании в определении $d_L$ участвует существенно большее число информационных символов в предшествующих последовательностях. Как известно [2], лучшие сверточные ко-

ды способны исправлять пакеты ошибок (или количество ошибок), длина которых численно равна половине количества проверочных символов на длине кодового блока.

При этом уравнения (3.13) упрощаются и принимают вид

$$
\begin{aligned}
S_k &= e_k^{\,u} + e_k^{\,p},\\
S_{k+7} &= e^u_{k+7} + e_k^{\,u} + e^p_{k+7},\\
S_{k+27} &= e^u_{k+20} + e^u_{k+27} + e_k^{\,u} + e^p_{k+27},\\
S_{k+76} &= e^u_{k+49} + e^u_{k+69} + e^u_{k+76} + e_k^{\,u} + e^p_{k+76},\\
S_{k+113} &= e^u_{k+37} + \ldots e_k^{\,u} + e^p_{k+113},\\
S_{k+137} &= e^u_{k+24} + e^u_{k+71} + \ldots e_k^{\,u} + e^p_{k+137},\\
S_{k+155} &= e^u_{k+18} + e^u_{k+42} + \ldots e_k^{\,u} + e^p_{k+155},\\
S_{k+156} &= e^u_{k+1} + e^u_{k+19} + \ldots e_k^{\,u} + e^p_{k-156},\\
S_{k+170} &= e^u_{k+14} + \ldots e^u_{k+170} + e_k^{\,u} + e^p_{k+170},\\
S_{k+202} &= e^u_{k+32} + \ldots e^u_{k+202} + e_k^{\,u} + e^p_{k+202}.
\end{aligned}
\qquad (3.21)
$$

Кроме самоортогональных свёрточных кодов пороговое декодирование допускает широкий класс сверточных кодов, полученных методом проб и ошибок с помощью ЭВМ, например, ПО-коды [11]. Для получения $J$ ортогональных проверок ПО-коды обычно требуют сложения нескольких символов синдрома ошибки, но при этом иногда получаются существенно лучшие коды (необходимое число проверок получается на меньшей длине кодового ограничения) особенно при декодировании с обратной связью.

При пороговом декодировании вероятность ошибки декодирования $P_д$ минимальна, если используется правило решения по максимуму апостериорной вероятности (правило МАВ). Для двоичного симметричного канала с вероятностью ошибки $p$ эта вероятность определяется неравенством, полученным в [11]:

$$
\mathrm{P}_д > 0{,}5\left(\frac{p}{1-p}\right)^{\frac{2p+n_0}{2p}} \qquad (3.22)
$$

В алгоритме декодирования по правилу МАВ (в некоторых источниках [8] это APP-алгоритм) неравенство (3.20) принимает вид

$$
\sum_{l=1}^{J} A_l W_l > T, \text{ то } e_k = 1, \qquad (3.23)
$$

где $W_l$ – весовые коэффициенты, пропорциональные надежности $l$-й проверки,

$$T = \frac{1}{2}\sum_{l=0}^{J} W_l . \qquad (3.24)$$

В [11] показано, что правило МАВ требует, чтобы

$$W_l = 2\log\left(\frac{1-p_l}{p_l}\right), \qquad (3.25)$$

где $p_l$ – вероятность того, что в проверке $A_l$ нечетное число символов (кроме $e^u_k$) принимает значение 1;

$$W_0 = 2\log\left(\frac{1-p}{p}\right), p_l = 0{,}5\left[1-\prod_{j=1}^{n_l}\left(1-2p_j\right)\right], \qquad (3.26)$$

$p_j$ – вероятность появления единицы в последовательности $e_1$, $e_2$, ... … $e_k$, ... $e_{n_l}$ в предположении, что элементы последовательности искажаются независимо; $n_l$ – объем проверки $A_l$.

В частном случае когда канал является стационарным или стационарным на длине кодового ограничения, $p_j = p$, а

$$p_l = 0{,}5\left[1-(1-2p)^{n_l}\right]. \qquad (3.27)$$

В каналах с переменными параметрами весовые коэффициенты являются функциями времени. Каждая величина $W_l$ – нелинейная функция объема проверки $n_l$ и вероятности ошибок декодируемых символов $p_j$, которая, в свою очередь, зависит только от уровня сигнала на выходе демодулятора, осуществляющего мягкое решение. Реализация порогового декодера в этом случае существенно усложняется. Веса $W_l$ могут быть вычислены с помощью либо нелинейного отображения с использованием аналоговых регистров сдвига, либо с приближенного метода APP [8], использующего цифровое представление сигнала на выходе демодулятора и цифровые регистры сдвига. Практические исследования показывают, что для получения результатов, близких к оптимальному АВ-декодированию, достаточно восьми уровней квантования.

Однако необходимо отметить возможность упрощения рассмотренного выше алгоритма в каналах со стираниями (трехуровневый канал). В этом случае все проверочные уравнения, содержащие

стёртые символы, не учитываются в системе проверок $\{A_l\}$, решение принимается только по оставшимся проверкам, веса которых приравниваются к единице. Возможные при этом ошибочные решения уточняются путем итераций.

Повторное декодирование (итерация) позволяет уменьшить вероятность ошибки декодирования. При этом для итераций может использоваться тот же декодер или несколько декодеров (многоступенчатый декодер) в зависимости от необходимой скорости работы декодера в канале. В первом случае декодер обычно называют итерационным, а во втором – многоступенчатым [14]. Физическая реализация и конкретный анализ алгоритмов итерационного и многоступенчатых декодеров будут рассмотрены позднее.

Недостатком декодирования с обратной связью является эффект распространения (размножения) ошибок в некоторых пороговых декодерах сверточных кодов. Размножение ошибок наблюдается при ухудшении качества канала, когда происходит засорение анализатора синдрома ложными коррекциями по цепи обратной связи из-за неправильных решений декодера в канале с большим уровнем помех. Эффект размножения ошибок слабо проявляется в мягком декодере и в большей степени – в жестком декодере. Для ослабления эффекта размножения ошибок необходимы соответствующий выбор кода и применение специальных мер, уменьшающих вероятность неправильных решений декодера или ослабляющих это влияние. Этой проблеме уделяется много внимания в специальной литературе [6, 8, 14, 15 и др.], однако и до настоящего времени она является актуальной. Возможные пути модификации пороговых декодеров, позволяющих ослабить эффект размножения ошибок и, следовательно, расширить способность декодера исправлять ошибки в каналах с большим уровнем помех, рассматриваются ниже.

Чем меньше плотность проверок (меньше проверок на длине кодового ограничения), тем в меньшей степени проявляется эффект размножения ошибок и пороговый декодер может нормально работать при большей плотности ошибок на входе. Для реализации корректирующей способности сверточного кода при малой плотности проверок необходимо многократное декодирование (итерации).

Практически необходимое число итераций при заданной вероятности ошибки декодирования зависит от выбора кода, алгоритма итераций и параметров декодера на каждой итерации.

В итерационных пороговых декодерах вероятность ошибки декодирования, по крайней мере, при достаточно малой вероятности ошибок на входе стремится к нулю. Поэтому жесткий пороговый декодер с итерациями при соответствующем выборе параметров кода и декодеров может быть альтернативой мягкому пороговому декодеру, тем более что мягкий декодер обычно эффективен только при полной априорной информации о статистических характеристиках сигналов и помех в канале связи.

Найдем оценку вероятности ошибки декодирования в итерационном декодере на $i$-й итерации. Эта вероятность может быть определена, если известно правило решения

$$\begin{aligned} P_{дк}^{(i)} &= M\left\{ P_{дк}^{(i-1)} \cdot P\left[ \left( \sum_{l=1}^{J} W_l A_l \le T \right) / (e_k^u = 1) \right] \right\} + \\ &\quad + M\left\{ (1 - P_{дк}^{(i-1)}) \cdot P\left[ \left( \sum_{l=1}^{J} W_l A_l > T \right) / (e_k^u = 0) \right] \right\} = \\ &= P_{дк}^{(i-1)} - M\left\{ P_{дк}^{(i-1)} \cdot P\left[ \left( \sum_{l=1}^{J} W_l A_l > T \right) / (e_k^u = 1) \right] \right\} + \\ &\quad + M\left\{ (1 - P_{дк}^{(i-1)}) \cdot P\left[ \left( \sum_{l=1}^{J} W_l A_l > T \right) / (e_k^u = 0) \right] \right\}, \end{aligned} \tag{3.28}$$

где $M\{*\}$ означает необходимость усреднения по всем значениям $W_l$ в канале с переменными параметрами; в канале с постоянными параметрами $W_l$ не изменяются.

Первый член суммы (3.28) определяет вероятность пропуска ошибочного символа ($e_k^u = 1$), второй – вероятность обнаружения (и исправления) ошибки, когда в канале ошибки не было ($e_k^u = 0$).

В тех случаях, когда канал стационарен на интервале, превышающем кодовое ограничение, можно считать, что на интервале стационарности при $i = 0$, $P_д^{(i)} = p$ – вероятности ошибки в канале

$$P_{дк}^{(i)} = P_д^{(i)}, \text{ а}$$

$$W_l = 2\log\left(\frac{1-p_l^{(i)}}{p_l^{(i)}}\right) = 2\log\frac{\left[1+\left(1-2P_\partial^{(i)}\right)^{n_l}\right]}{\left[1-\left(1-2P_\partial^{(i)}\right)^{n_l}\right]}. \qquad (3.29)$$

Следовательно, $W_l = f(P_\partial^{(i)}, n_l, i,)$, то есть коэффициенты должны изменяться (увеличиваться) с увеличением номера итерации. Заметим, что увеличение $W_l$ можно компенсировать соответствующим уменьшением (увеличением) уровня порога $T$. Для определения направления изменения $T$ необходимо преобразовать выражение (3.28) с учетом того, что условная вероятность необнаруженной ошибки

$$P\left[(\sum_{l=1}^{J} W_l A_l \le T)/(e_k^u = 1)\right] = P\left[\sum_{l=1}^{J} W_l A_l \ge \left(\sum_{l=1}^{J} W_l - T\right)/(e_k^u = 0)\right], \qquad (3.30)$$

так как $\sum_{l=1}^{J} W_l$ – это общая взвешенная сумма проверок, для которых $A_l = 1$.

Тогда

$$\begin{aligned} P_{\partial к}^{(i)} &= P_{\partial к}^{(i-1)} \cdot P\left[\sum_{l=1}^{J} W_l A_l \ge \left(\sum_{l=1}^{J} W_l - T\right)/(e_k^u = 0)\right] + \\ &+ (1 - P_{\partial к}^{(i-1)}) \cdot P\left[\left(\sum_{l=1}^{J} W_l A_l > T\right)/(e_k^u = 0)\right] = \\ &= P_{\partial к}^{(i-1)} - \left\{P_{\partial к}^{(i-1)} \cdot P\left[\sum_{l=1}^{J} W_l A_l < \left(\sum_{l=1}^{J} W_l - T\right)/(e_k^u = 0)\right]\right\} + \\ &+ \left\{(1 - P_{\partial к}^{(i-1)}) \cdot P\left[\left(\sum_{l=1}^{J} W_l A_l > T\right)/(e_k^u = 0)\right]\right\}. \end{aligned} \qquad (3.31)$$

Используя (3.31), можно найти условия, при которых $P_{\partial к}^{(i-1)} > P_{\partial к}^{(i)}$, то есть итерации уменьшают вероятность ошибки на выходе декодера.

$$\left\{P_{\partial к}^{(i-1)} \cdot P\left[\sum_{l=1}^{J} W_l A_l > T / e_k^u = 1\right] - \left(1 - P_{\partial к}^{(i-1)}\right) \cdot P\left[\sum_{l=1}^{J} W_l A_l > T / e_k^u = 0\right]\right\} > 0. \qquad (3.32)$$

Вероятность того, что при $e_k^u = 1$ ошибка обнаружится и будет исправлена, равна вероятности четного числа "1" в ошибочных символах (кроме $e_k^u$ ) $l$-го проверочного уравнения.

$$g_l^{(i)} = \frac{1+\left(1-2P_{\partial к}^{(i-1)}\right)^{n_l}}{2},$$

а вероятность того, что при $e_k^u = 0$ ошибка обнаружится и будет принято неверное решение об исправлении ошибки, равна вероятности нечетного числа "1", в $l$-м проверочном уравнении

$$p_l^{(i)} = \frac{1-\left(1-2P_{\partial к}^{(i-1)}\right)^{n_l}}{2} = 1 - g_l^{(i)}. \qquad (3.33)$$

Таким образом, в уравнениях (3.28) и (3.32) речь идет о суммировании независимых случайных величин, вероятности которых равны соответственно $p_l^{(i)}$ и $g_l^{(i)} = 1 - p_l^{(i)}$.

Суммирование затруднено тем, что в общем случае неизвестно распределение вероятностей этих величин, а применение центральной предельной теоремы теории вероятностей невозможно из-за небольшого (чаще всего) числа суммируемых величин и большого отличия значений этих величин. Для решения аналогичных уравнений обычно используется метод производящих функций [8].

Для двоичного симметричного канала производящая функция для произведения ($W_l \cdot A_l$) равна

$$G_l(x) = p_l x^{W_l} + g_l. \qquad (3.34)$$

Производящая функция $\sum_{l=1}^{J} W_l A_l$ равна произведению $G_l(x)$

$$G(x) = \prod_{l=1}^{J} (p_l x^{wl} + g_l) = \sum_{l=1}^{J} G_l x^l. \qquad (3.35)$$

Тогда неравенство (3.32) можно переписать в виде

$$P_{\partial к}^{(i-1)}\left(\sum_{l>T}^{J} G_l^{(i)} / (e_k^u = 1)\right) - \left(1 - P_{\partial к}^{(i-1)}\right)\left(\sum_{l>T}^{J} G_l^{(i)} / (e_k^u = 0)\right) > 0, \qquad (3.36)$$

где суммируются $G_l$ -коэффициенты в $G(x)$ тех членов производящей функции, у которых степени при $x$ больше $T$, так как каждый коэффициент в производящей функции – это вероятность того, что рассматриваемая случайная величина равна показателю при $x$. При этом вероятность ошибки декодирования равна

$$P_{\partial к}^{(i)} = P_{\partial к}^{(i-1)} - P_{\partial к}^{(i-1)}\left(\sum_{l>T}^{J} G_l^{(i)} / (e_k^u = 1)\right) + \left(1 - P_{\partial к}^{(i-1)}\right)\left(\sum_{l>T}^{J} G_l^{(i)} / (e_k^u = 0)\right) > 0. \qquad (3.37)$$

Коэффициенты $G_l^{(i)}/(e_k^u = 1)$ и $G_l^{(i)}/(e_k^u = 0)$ отличаются тем, что в производящей функции (3.34) $p_l$ и $g_l$ меняются местами.

Выражение (3.37) можно существенно упростить, если принять, что декодер является жестким и все $W_l = 1$ (как будет показано ниже, жесткий декодер с итерациями может быть альтернативой мягкому декодеру).

Тогда

$$(G_l^{(i)}/(e_k^u = 1)) = C_J^l \left(1 - p_l^{(i)}\right)^l \left(p_l^{(i)}\right)^{J-l}, \qquad (3.38)$$
$$(G_l^{(i)}/(e_k^u = 0)) = C_J^l \left(1 - p_l^{(i)}\right)^{J-l} \left(p_l^{(i)}\right)^l.$$

В этом случае

$$P_\partial^{(i)} = P_\partial^{(i-1)} - P_\partial^{(i-1)} \sum_{l=T+1}^{J} C_J^l \left(1 - p_l^{(i)}\right)^l \left(p_l^{(i)}\right)^{J-l} + \qquad (3.39)$$
$$+ \left(1 - P_\partial^{(i-1)}\right) \sum_{l=T+1}^{J} C_J^l \left(p_l^{(i)}\right)^l \left(1 - p_l^{(i)}\right)^{J-l},$$

а $P_\partial^{(i)} < P_\partial^{(i-1)}$, если выполняется неравенство

$$P_\partial^{(i-1)} \sum_{l=T+1}^{J} C_J^l \left(1 - p_l^{(i)}\right)^l \left(p_l^{(i)}\right)^{J-l} - \left(1 - P_\partial^{(i-1)}\right) \sum_{l=T+1}^{J} C_J^l \left(p_l^{(i)}\right)^l \left(1 - p_l^{(i)}\right)^{J-l} > 0.$$

После преобразования получаем

$$\sum_{l=T+1}^{J} \frac{\left(1 - p_l^{(i)}\right)^l \left(p_l^{(i)}\right)^{J-l}}{\left(p_l^{(i)}\right)^l \left(1 - p_l^{(i)}\right)^{J-l}} > \frac{1 - P_\partial^{(i-1)}}{P_\partial^{(i-1)}}; \quad \sum_{l=T+1}^{J} \left[\frac{\left(1 - p_l^{(i)}\right)^l}{\left(p_l^{(i)}\right)^l}\right]^{2l-J} > \frac{1 - P_\partial^{(i-1)}}{P_\partial^{(i-1)}}.$$

Следовательно, с уменьшением $P_l^{(i)}$, а значит, и вероятности ошибки в канале связи $p$ уровень порога $T$ можно уменьшать. Очевидно, что $\min \{P_{\partial к}^{(i)}\}$ будет иметь место в том случае, если первый член суммы удовлетворяет неравенству

$$\left[\frac{1 - p_l^{(i)}}{p_l^{(i)}}\right]^{2(T+1)-J} > \frac{1 - P_\partial^{(i-1)}}{P_\partial^{(i-1)}}. \qquad (3.40)$$

Учитывая, что $P_\partial^{(i-1)} < p_l^{(i)} << 1$ и $[2(T + 1) - J] > 1$,

$$T > \frac{J-1}{2}. \qquad (3.41)$$

Неравенство (3.41) показывает, что в пороговом декодере значение порога может быть (при достаточно малых вероятностях ошибки в канале) даже меньше, чем это требует мажоритарное правило решения, согласно которому $T = J/2$. Однако с учетом того, что $T$ является ближайшим целым числом, при малых $J$ значения порога будут одинаковыми.

С другой стороны, на первых итерациях, когда вероятность ошибки достаточно велика, уровень порога надо устанавливать больше $T > J/2$, что позволит уменьшить $P_д$ для следующей итерации.

В таком подходе к итерационному декодированию имеется и более глубокий смысл, заключающийся в том, что увеличение порога на первых итерациях позволит увеличить исходную вероятность ошибки $p$ в канале связи, при которой еще не проявляется эффект размножения ошибок.

На рис. 3.6 показан характер зависимости вероятности ошибки декодирования $P_д$ от вероятности ошибок в стационарном канале $p$. Численные значения, рассчитанные для сверточного кода $R$=1/2 по формуле (3.39) для порогового декодера и для АВ-декодера (одна итерация), приведены в табл. 3.2.

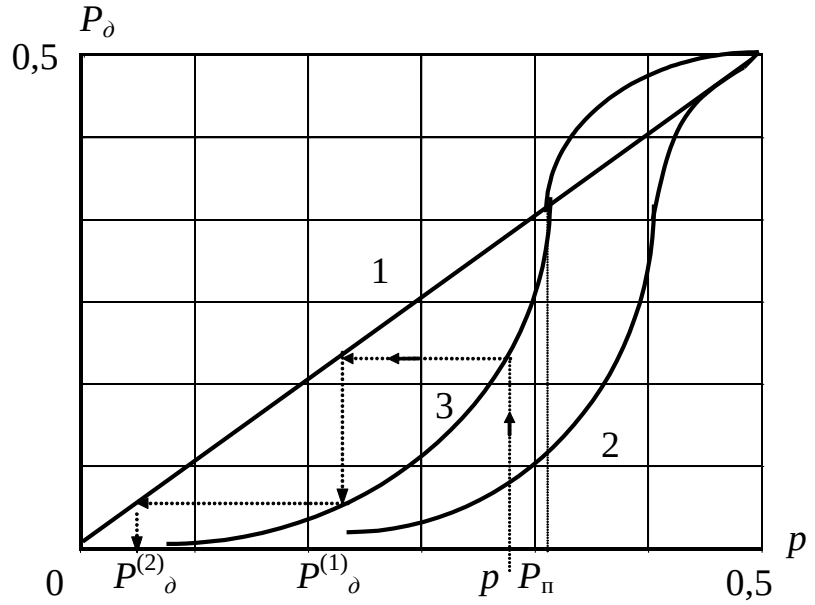



Рис. 3.6. Зависимость $P_{дк}=f(p)$: 1 - ДК без кодирования;
2 - АВ декодер; 3 - пороговый декодер

Из рис. 3.6 и табл. 3.2 очевидно, что пороговый декодер с жестким решением размножает ошибки, когда $p > P_П$, в отличие от

декодера АВ. Причем предельная вероятность $P_П$ зависит не только от параметров кода, но и от характеристик кода и декодера, в частности, от значения порога $T$.

Табл. 3.2. *Значения вероятности ошибок декодирования $P_д$ при различных уровнях порога*

| Свёрточный код (406,203), $R$=1/2, непрерывный декодер | | | | | |
|---|---|---|---|---|---|
| $p$ | $T$=9 | $T$=8 | $T$=7 | $T$=6 | $T$=5 |
| 0.4 | 0.401 | 0.403 | 0.404 | 0.41 | 0.415 |
| 0.3 | 0.3* | 0.302 | 0.31 | 0.32 | 0.335 |
| 0.2 | 0.2 | 0.202* | 0.21 | 0.22 | 0.25 |
| 0.1 | 0.0997 | 0.0995 | 0.106* | 0.115 | 0.151 |
| 0.075 | 0.0745 | 0.072 | 0.07 | 0.074* | 0.104 |
| 0.05 | 0.049 | 0.0484 | 0.036 | 0.033 | 0.0268 |
| 0.04 | 0.039 | 0.0318 | 0.02 | 0.007 | 0.0068 |
| 0.03 | 0.027 | 0.0203 | 0.0091 | 0.0023 | 0.0012 |
| 0.02 | 0.016 | 0.01 | 0.0018 | 0.00033 | 0.0002 |
| 0.01 | 0.006 | 0.0015 | 0.00014 | 0.0000148 | 0.0000015 |
| 0.001 | 0.000079 | 0.000005 | - | - | - |

Рис. 3.6 можно использовать для доказательства того, что вероятность ошибки декодирования порогового декодера стремится к нулю при увеличении числа итераций, если

$$0 < P_д^{(i)} < P_д^{(i-1)} < p < P_п, \qquad (3.42)$$

Процесс итераций показан на рис. 3.6 пунктиром. Процесс итераций сходится согласно критерию сходимости Коши в точке [0, 0], когда $p < P_П$, и в точке [0,5; 0,5], когда $p > P_П$, так как $P_д^{(i)} < P_д^{(i-1)}$ при $p < P_П$ и $P_д^{(i)} > P_д^{(i-1)}$ при $p > P_П$; причём точка $[P_П, P_П]$ является точкой неустойчивого равновесия.

Теоретические и экспериментальные исследования показывают, что при $p << P_{\text{П}}$ скорость сходимости $P_{\partial к}^{(i)}$ мало отличается в АВ- декодере и жестком пороговом декодере, достаточно 2 – 3 итераций. Однако если канал плохого качества и $p$ близко к 0,5, даже в АВ-декодере требуется большее число итераций.

Таким образом, если $p << P_{\text{П}}$, пороговый декодер (в дальнейшем пороговый декодер с жестким решением будем называть просто пороговым декодером) с итерациями может обеспечить декодирование не хуже декодера АВ.

Вероятность $P_{\text{П}}$ можно вычислить из условия $P_{\partial}^{(i)} = P_{\partial}^{(i-1)} = P_{\text{П}}$, используя (3.30) и (3.39):

$$P_{\text{П}} = \frac{P\left[(\sum_{l=1}^{J} W_l A_l > T)/(e_k^u = 0)\right]}{P\left[(\sum_{l=1}^{J} W_l A_l > T)/(e_k^u = 0)\right] + P\left[\sum_{l=1}^{J} W_l A_l \geq (\sum_{l=1}^{J} W_l - T)/(e_k^u = 0)\right]} = \tag{3.43}$$

$$= \frac{\sum_{l=T+1}^{J} C_J^l \left(p_l^{(i)}\right)^l \left(1-p_l^{(i)}\right)^{J-l}}{\sum_{l=T+1}^{J} C_J^l \left(p_l^{(i)}\right)^l \left(1-p_l^{(i)}\right)^{J-l} + \sum_{l=T+1}^{J} C_J^l \left(1-p_l^{(i)}\right)^l \left(p_l^{(i)}\right)^{J-l}}.$$

Согласно (3.43) значение $P_{\text{П}}$ можно увеличить, если уменьшать количество ортогональных проверок $J$ (использовать коды с малым числом проверок) или увеличивать порог $T$. На рис.3.7 и 3.8 приведены результаты расчётов значений $P_{\text{П}}$ в зависимости от $J$ для ряда кодов и в зависимости от $T$ для свёрточного кода (406,203).

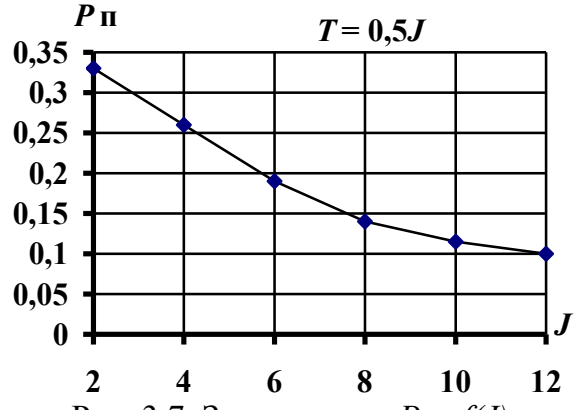


Рис. 3.7. Зависимость $P_{\text{п}}=f(J)$

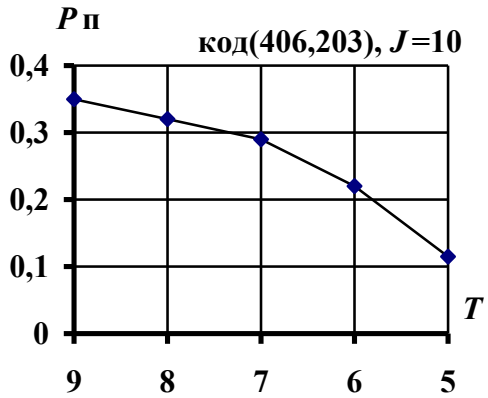


Рис. 3.8. Зависимость $P_{\text{п}}=f(T)$

Причем если уменьшение $J$ приводит к снижению динамики итераций (требуется все большее число итераций для достижения заданного $P_{\partial}^{(i)}$), то регулированием $T$ можно добиться лучших результатов. Увеличивать порог $T$ целесообразно, когда вероятность ошибки на входе декодера (или данной ступени итерации) близка к $P_{\text{П}}$; после того как $P_{\partial}^{(i)}$ становится существенно меньше $P_{\text{П}}$, порог может быть восстановлен до $T = 0{,}5\,J$.

Необходимо также учитывать, что в жестком пороговом декодере в отличие от оптимального порогового декодера по максимуму апостериорной вероятности (АВ-декодера) [11], при $p \geq P_{\text{П}}$ проявляется эффект размножения ошибок, причем его влияние почти не устраняется итерациями. Поскольку в пороговом декодере $P_{\text{П}} < 0{,}5$, всегда имеется вероятность того, что на отдельных участках декодируемой последовательности (на интервале кодового ограничения) локальная вероятность ошибки превысит $P_{\text{П}}$ и в результате эффекта размножения появятся пакеты ошибок, которые не могут быть исправлены на последующих итерациях. Следовательно, поток независимых ошибок на входе декодера, преобразуется в поток пакетирующихся ошибок на выходе. Причем в хорошем канале эти пакеты будут более редкими, чем в плохом канале. Отличие декодера АВ и порогового декодера в данном случае в том, что с увеличением числа итераций в декодере АВ пакеты ошибок будут постепенно исчезать, а в пороговом декодере с жесткими решениями некоторые пакеты будут уплотняться (в системах с обратной связью по этому признаку можно формировать сигнал переспроса). Это приведет к увеличению средней вероятности ошибки декодирования на некоторую величину $\Delta P$, к которой будет стремиться $P_{\partial к}$ с увеличением числа итераций. Уменьшить влияние этого эффекта на вероятность ошибки декодирования можно путем оптимизации уровня порога на каждой итерации, особенно на первых итерациях.

Грубая оценка вероятности $\Delta P$ может быть получена, если принять во внимание, что наиболее вероятно такое событие на первой итерации

$$\Delta P \approx \left(\frac{k}{n-k}\right) \sum_{v=P_{II}\cdot(n-k)}^{(n-k)} vP(v), \qquad (3.44)$$

где $P(v)$ – вероятность того, что в последовательности из $n$ символов $v$ ошибок; $k$ – число информационных символов в последовательности $n$.

Для симметричного двоичного канала с постоянными параметрами $P(v)$ можно определить по биномиальной формуле

$$P(v) = C_{n-k}^{v} p^{v} \left(1 - p\right)^{n-k-v}.$$

В то же время очевидно, что $0 < \Delta P < p$, так как ошибки, определяемые формулой (3.44), являются лишь частью общего числа ошибок на выходе декодера после первой итерации.

Более точная оценка вероятности ошибки декодирования для итерационного декодера может быть получена по результатам статистических испытаний модели декодера на ЭВМ, так как эта оценка определяется как свойствами канала, так и характеристиками конкретного кодера.

Чтобы уменьшить вероятность $\Delta P$, необходимо принять меры к увеличению $P_{\text{П}}$ и соответствующим образом добиваться, чтобы вероятность ошибки в канале на первой итерации $p$ была не выше допустимой. В каналах с пакетирующимися ошибками уменьшение $p$ на интервале кодового ограничения может быть достигнуто за счет применения перемежения символов канала.

На рис. 3.9 показаны зависимости $P_{\partial} = f(p)$ в области больших значений $p$, которые позволяют проследить влияние изменения порога в итерационном декодере (кривые 2 и 3) на изменение $P_{\text{П}}$.

Видно, что изменение порога уменьшает $P_{\text{П}}$ (вероятность при которой $P_{\partial} = p$) по сравнению не только с итерациями при постоянном пороге, но и с декодированием без итераций (кривая 1). Кроме того, можно заметить, что итерации с переменным порогом почти не ухудшают качество декодирования в области значений $p > P_{\text{П}}$, что характерно для итераций с постоянным порогом.

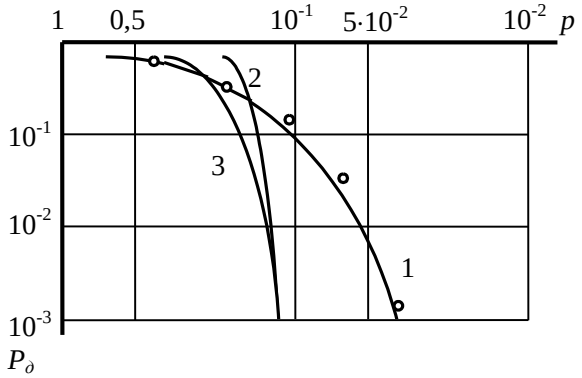



Рис. 3.9. Зависимость $P_д=f(p)$: 1 - одна итерация; 2 - пять итераций при $T=J/2$; 3 - пять итераций при $T_{1\text{-}5}=9\div5$; O- одна итерация

Зависимость ЭВК=$f(p)$ для АВ-декодера, итерационного модифицированного порогового алгоритма и порогового алгоритма декодирования без итераций показана на рис. 3.10. Расчеты выполнены для когерентного приема ЧМ-сигнала. Для итерационного алгоритма учтена поправка $\Delta P$, рассчитанная по формуле (3.36) для рассматриваемого кода.

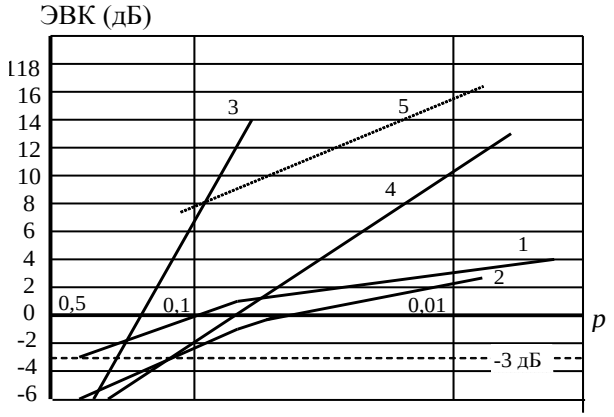



Рис. 3.10. Зависимость ЭВК от вероятности ошибки в канале для декодеров: 1 - АВ; 2 - порогового; 3 - порогового, пять итераций; 4 - порогового, пять итераций с поправкой $\Delta P$; 5 - теоретический предел

Видно, что поправка существенно уменьшает предельную вероятность $P_П$ и уменьшает выигрыш от итераций. В области относительно малых вероятностей $p$ поправка $\Delta P$ фактически определяет предельно достижимое качество декодирования в итерационном пороговом декодере. При этом ЭВК значительно больше, чем в декодерах АВ и пороговом декодере без итераций. Кривая 5 на рис. 3.10 показывает предельно достижимый выигрыш двоичных сверточных кодов при $R$=0,5, допускающих пороговое декодирование. Полученные результаты не противоречат этому пределу.

## ЗАДАЧИ

1. По каналу связи с нормальным белым шумом передается информация со скоростью $V_{пи}$ = **10 кбит/с**. При этом средняя вероятность ошибок в канале составляет $P_{ош} = 10^{-6}$.

Для улучшения качества передачи информации рассматривается несколько вариантов решения:

– кодирование корректирующим кодом с порождающей матрицей $G = |\ 111\ |$;

– кодирование сверточным (6,3)-кодом;

– кодирование (8,4)-кодом Хемминга;

– уменьшение на 20 % скорости передачи.

Какое из предложенных решений обеспечит больший эффект, если в первых трех случаях скорость передачи информации должна сохраниться прежней?

2. Порождающая матрица линейного блочного кода имеет вид

$$\left|\begin{matrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{matrix}\right| .$$

Определить для данного кода $H$, $d_{\min}$. Изобразить схему кодера и декодера.

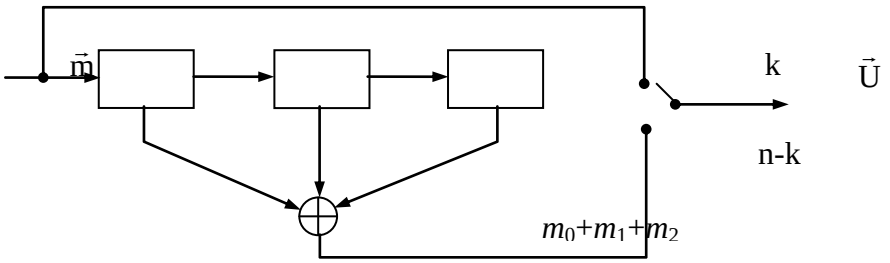
Определить число ошибок, не исправленных данным кодом за 1 ч работы, если $V_{пи}$ = **10 кбит/с**, $P_{ош} = 10^{-4}$.

3. В канале связи с шумами производится кодирование информации с использованием кода с порождающей матрицей вида $G = |\ 11111\ |$. Постройте проверочную матрицу **H**, вычислите кодовое расстояние и определите корректирующие способности кода.

4. Для кодера сверточного кода со схемой, показанной на рис. 2.5, определить $k_0$, $n_0$, $m$, $n$, $k$, $R$, $b$, изобразить кодовое дерево и решетчатую диаграмму, закодировать последовательность $m$ = (**1100100000...** , декодировать $r$ с ошибкой во втором кадре с использованием алгоритма Витерби.

5. Предложить вариант схем кодера и декодера сверточного (9.6)-кода Вайнера – Эша (по аналогии с (12,9)-кодом), исправляющего одиночную ошибку на сегменте из трех кодовых кадров. Проиллюстрировать работу кодера и декодера на примере.

6. Схема кодера линейного блочного кода приведена на рисунке. Найти для него ***H, G,*** $d_{\min}$, $P_{но}$ , $P_{ни}$. Изобразить схему синдромного декодера.




7. Для сверточного кода со схемой рис. 2.5 ((6,3)-код) определить $d_{\min}$, закодировать последовательность ***m*** = (**1100000000...** , декодировать принятую последовательность с двойной ошибкой в третьем кадре с использованием алгоритма Витерби и Фано.

8. Изобразить схему кодера и декодера Меггитта для циклического (8,4)-кода. Привести пример кодирования и декодирования с одиночной ошибкой.

9. По каналу связи с шумами передается двоичная информация со скоростью $V_{ни}$ = **1 Мбит/с**. При этом в среднем 1 раз в минуту в канале возникает ошибка.

Для уменьшения частоты ошибок предложено использовать несколько вариантов кодирования:

- (7,4)-кодом Хемминга;
- сверточным (6,3)-кодом;
- кодом с порождающей матрицей вида ***G*** = | **111** |.

Какое из предложенных решений обеспечит больший эффект, если скорость передачи информации $V_{ни}$ должна остаться неизменной?

10. Предложить варианты схем кодера и декодера сверточного (15,12)-кода (по аналогии с (12,9)-кодом Вайнера – Эша, исправляющего одиночную ошибку на сегменте из трех кодовых кадров).

Привести пример кодирования и декодирования.

11. В цифровой двоичной системе связи информация передается со скоростью **2 Мбит/с**, при этом в среднем один раз в минуту в канале возникает ошибка.

Как изменится частота появления ошибок, если в канале производить кодирование сверточным (6,3)-кодом и для сохранения скорости передачи информации в 2 раза повысить скорость передачи двоичных символов?

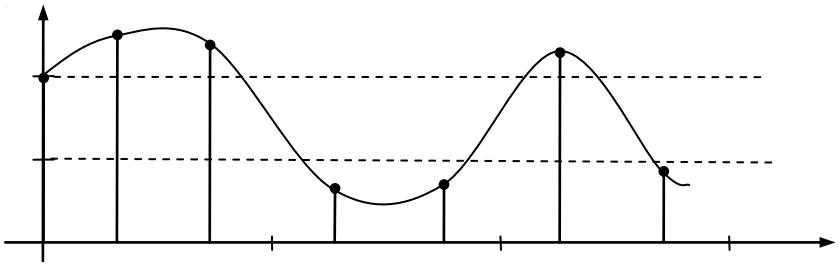
12. Изобразить схему декодера Меггита для циклического (7,3)-кода. Привести пример кодирования и декодирования с одиночной ошибкой.

13. Порождающая матрица блочного кода имеет вид $\boldsymbol{G}$ = | **111** |. Найти $\boldsymbol{H}$, $\boldsymbol{d_{min}}$, $\boldsymbol{P_{но}}$, $\boldsymbol{P_{ни}}$ данного кода. Изобразить схему кодера и декодера.

14. В канале связи с шумами производится кодирование информации с использованием блочного кода с порождающей матрицей $\boldsymbol{G}$ = | **11111** |. На входе приемного устройства присутствует колебание $\boldsymbol{U(t)}$ вида, показанного на рисунке. Сигнал $\boldsymbol{U(t)}$ подвергается дискретизации, причем на интервал длительностью в один символ приходится два отсчета $\boldsymbol{U(t)}$.

Какое решение относительно $\boldsymbol{m}$ вынесет по принятой реализации:

- мягкий декодер максимального правдоподобия;
- жесткий мажоритарный декодер?



15. Двоичный циклический код, заданный порождающим полиномом

$$\boldsymbol{g(x) = 1 + X^2 + X^3 + X^4},$$

позволяет исправлять пакеты ошибок длиной 2 (двойные ошибки в соседних символах).

Определить длину кода. Сконструировать декодер Меггита для данного кода.

16. Изобразить схему, построить кодовое дерево и решетчатую диаграмму для несистематического сверточного кода с $R = 1/3$, $m = 2$ и имеющего порождающие полиномы вида

$$g_1(x) = 1 + X + X^2,\ g_2(x) = 1 + X + X^2 \text{ и } g_3(x) = 1 + X^2.$$

17. По двоичному каналу связи передается информация со скоростью 9600 бит/с.

Сколько времени понадобится для передачи 1000 слов русского текста (энтропия $H(\lambda) = 2$ бит/букву) с использованием примитивного равномерного двоичного кода и кода без избыточности (одна страница – 2000 букв )?

18. Итеративный код задан матрицей вида

$$U = \begin{vmatrix} m0 & m1 & p0 \\ m2 & m3 & p1 \\ p2 & p3 & p4 \end{vmatrix}.$$

Записать порождающую матрицу эквивалентного ему линейного блочного систематического $(n,k)$-кода. Определить исправляющую способность кода, найти вероятность неисправления ошибки, если вероятность ошибок в канале составляет $P_{ош} = 10^{-4}$.

19. Итеративный код задан матрицей вида

$$U = \begin{vmatrix} m_0 & m_1 & m_2 & m_0 + m_1 & m_1 + m_2 \\ m_3 & m_4 & m_5 & m_3 + m_4 & m_4 + m_5 \\ m_6 & m_7 & m_8 & m_6 + m_7 & m_7 + m_8 \\ m_0 + m_3 & m_1 + m_4 & m_2 + m_5 & p_1 & p_2 \\ m_5 + m_6 & m_4 + m_7 & m_5 + m_8 & p_3 & p_4 \end{vmatrix}.$$

Проверочные символы $P_1 \ldots P_4$ формируются путем суммирования всех информационных символов, входящих в соответствующие столбцы и строки матрицы, например $p_1 = m_0 + m_1 + m_3 + m_4 + m_6 + m_7 + m_0 + m_3 + m_1 + m_4 + m_2 + m_5$.

Записать порождающую матрицу эквивалентного линейного блочного систематического ***(n,k)***-кода. Определить исправляющую способность кода. Найти вероятность не исправляемой данным кодом ошибки, если вероятность ошибки в канале составляет $P_{ош} = 10^{-3}$.

20. Исправляющий двойные ошибки циклический (15,7)-код БЧХ имеет порождающий полином вида

$$G(x) = X^8 + X^7 + X^6 + X^4 + 1.$$

Построить кодер и декодер Меггита для этого кода.

21. Дискретный источник выдает символы из ансамбля ***{ ai }*** объемом $K = 50$.

Какое минимальное число разрядов должен иметь равномерный двоичный код, предназначенный для кодирования символов данного ансамбля? Записать примеры кодовых слов. Какова избыточность примитивного кода, если энтропия источника составляет 3 бит/букву?

22. Ансамбль дискретных символов $\{ a_i \}$ объемом $K = 32$ имеет энтропию $H(A) = 2$ **бит/символ.**

Найти минимальное количество кодовых символов, которое надо израсходовать на кодирование символа источника равномерным примитивным двоичным кодом. Какое избыточное количество символов по сравнению с оптимальным кодом приходится использовать на один символ источника при примитивном кодировании?

23. Закодировать двоичным кодом Шеннона – Фано ансамбль $\{ a_i \}$, если вероятность символов имеет значения, приведенные ниже.

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ |
|---|---|---|---|---|---|---|---|
| 0.25 | 0.25 | 0.125 | 0.125 | 0.0625 | 0.0625 | 0.0625 | 0.0625 |

Найти среднее число символов кодовой комбинации. Определить избыточность кода.

24. В цифровой системе телевидения высокой четкости (ТВЧ) передача одного кадра изображения размерами 1500×1000 элементов с числом градаций яркости $M = 256$ производится за $T_к = 40$ **мс.**

Какую полосу частот будет занимать цифровой телевизионный сигнал при использовании примитивной КИМ? Как изменится эта величина, если степень корреляции соседних элементов изображения составляет 0.95 и производится кодирование с полным устранением избыточности?

25. Некоторый дискретный источник выдает независимые символы из ансамбля $\{ a_i \}$ ($i = 1,2,...9$) с вероятностями, определенными следующим образом:

| $a_1$ | $a_2$ | $a_3$ | $a_4$ | $a_5$ | $a_6$ | $a_7$ | $a_8$ | $a_9$ |
|---|---|---|---|---|---|---|---|---|
| 0.2 | 0.15 | 0.15 | 0.12 | 0.1 | 0.1 | 0.08 | 0.06 | 0.04 |

Закодировать символы данного ансамбля кодом Хаффмена. Построить кодовое дерево и определить среднюю длину кодового слова.

26. Циклический (15,4)-код задан порождающим полиномом вида

$$g(x) = X^{11} + X^8 + X^5 + X^3 + X^2 + X + 1.$$

Построить кодер и декодер Меггита для этого кода. Определить минимальное хеммингово расстояние и исправляющую способность кода. Найти вероятность неисправления ошибки, если вероятность ошибки в канале составляет $P_{ош} = 10^{-5}$.

27. Показать, что хороший декодер линейного блочного кода должен производить нелинейные операции, для чего доказать:

а) что процедура вычисления синдрома линейна по отношению к вектору ошибок, т.е. если $S = F(e)$, то $F(e_1 + e_2) = F(e_1) + F(e_2)$;

б) линейный – это такой декодер, у которого функция $e = f(S)$, связывающая синдром и оценку вектора ошибок, удовлетворяет условию $f(S_1 + S_2) = f(S_1) + f(S_2)$;

в) если мы хотим, чтобы декодер исправлял все одиночные ошибки, то функция $e = f(S)$, связывающая синдром и оценку вектора ошибок, должна быть нелинейной.

Доказать, что линейный декодер может исправлять не более $n-k$ из $n$ возможных одиночных ошибок.

28. Полиномиальный (17,9)-код задан порождающим многочленом вида

$$g(x) = X^8 + X^5 + X^4 + X^3 + 1.$$

Определить минимальное расстояние Хемминга для данного кода. Сколько ошибок может исправить этот код? Построить кодер и декодер Меггита для данного кода. Определить вероятность не исправляемой кодом ошибки, если вероятность ошибки в канале составляет $P_{ош} = 10^{-3}$.

29. Кодом с проверкой на четность называется код, который образуется путем добавления к $k$-разрядной информационной последовательности одного символа так, чтобы число единиц в полученном коде было четно.

Построить кодер и декодер для (8,7)-кода с проверкой на четность. Определить вероятность необнаруживаемой ошибки, если вероятность ошибки приема символа составляет $P_{ош} = 10^{-3}$. Изобразить схемы кодера и декодера.

30. Исправляющий три ошибки (23,12)-код является циклическим с порождающим полиномом

$$G(x) = X^{11} + X^{10} + X^6 + X^5 + X^4 + X^2 + 1,$$

или $$G(x) = X^{11} + X^9 + X^7 + X^6 + X^5 + X + 1.$$

Найти проверочный многочлен $h(x)$ для данного кода. Построить кодер на основе $g(x)$. Построить декодер Меггита для данного кода.

31. Рассмотреть линейный блочный код, кодовое слово которого формируется по правилу

$$U=(x_0,\ x_1,\ x_2,\ x_3,\ x_4,\ x_0+x_1+x_2+x_3+x_4,\ x_0+x_2+x_3+x_4,\ x_0+x_1+x_2+x_4,\ x_0+x_1+x_2+x_3\ ).$$

Найти проверочную матрицу кода и параметры $n, k$ .

32. Добавить к коду из предыдущей задачи общую проверку на четность и построить соответствующую проверочную матрицу.

Чему равно минимальное кодовое расстояние полученного кода?

33. Построить для кодов из предыдущих задач порождающие матрицы по проверочным.

34. Двоичный код, предназначенный для кодирования сообщений источника с алфавитом $M = 8$, содержит следующие кодовые слова:

$$U_1=00000;\ U_2=10011;\ U_3=01010;\ U_4=11001;$$
$$U_5=00101;\ U_6=10110;\ U_7=01111;\ U_8=11100.$$

Является ли данный код линейным и систематическим?

Определить возможности кода по обнаружению и исправлению ошибок.

Если код является линейным, построить порождающую и проверочную матрицы кода, схемы кодирования и декодирования.

35. Спроектировать блоки кодирования и декодирования данных для системы передачи информации.

Исходные данные:

– источник выдает информацию блоками по 4 бита;
– производительность источника 0,4 Мбит/с;
– используется (7,3)-код Хемминга;
– затухание сигнала на трассе $D$ = **150 дБ**;
– коэффициент усиления передающей и приемной антенн $G$ = **32**;
– чувствительность приемника радиолинии $N_0 = 10^{-18}$ **Вт/Гц**;
– прием сообщения посимвольный с использованием ортогональных сигналов.

Определить мощность передатчика системы связи, обеспечивающую вероятность безошибочного приема блока сообщения $P$ **= 0,999999.**

36. Одним из способов улучшения корректирующих свойств кодов является добавление общей проверки на четность (если число единиц в кодовом слове нечетно, добавляется 1 , если четно – 0), что эквивалентно перекодированию кодовых слов следующим образом:

$$U_1 = m \times G_1, \qquad U_2 = U_1 \times G_2.$$

Записать выражение для перекодирующей матрицы $G_2$, соответствующей исходному (7,3)-коду Хемминга.

Записать выражение для проверочной матрицы нового кода. Изобразить схемы кодирующего и декодирующего устройств.

Как изменится вероятность необнаружения ошибки $P(E)$ в сравнении с исходным (7,3)-кодом, если вероятность ошибки в канале $P_{ош} = 10^{-5}$?

37. По каналу связи передается информация со скоростью $V$ = **2 кбит/с.** Используются двоичные сигналы типа 1 и –1. Мощность сигнала на входе приемника составляет $P = 10^{-14}$ **Вт.** Спектральная плотность мощности помех, приведенная ко входу, $N_0 = 10^{-18}$ **Вт/Гц.** При передаче используется корректирующий (7,3)-код Хемминга.

Определить число необнаруживаемых и неисправленных ошибок, проходящих по каналу связи за один час работы.

38. Для кодирования информации в системе связи используется (7,3)-код Хемминга. Скорость передачи 1 кбит/с. В канале связи действует нормальная "белая" помеха со спектральной плотностью $N_0 = 10^{-18}$ **Вт/Гц**.

При какой мощности сигнала на входе приемника вероятность неисправленной ошибки составит $P_{ни} = 10^{-8}$ ?

39. Порождающая матрица кода имеет вид

$$G = \begin{vmatrix} 1\,0\,0\,1\,0\,0\,1 \\ 0\,1\,0\,0\,1\,0\,1 \\ 0\,0\,1\,0\,0\,1\,1 \end{vmatrix}.$$

Найти проверочную матрицу кода; изобразить схемы кодирующего и декодирующего устройств; найти минимальное кодовое расстояние кода; определить возможности кода по обнаружению и исправлению ошибок.

Определить вероятность пропуска необнаруживаемой ошибки $P(E)$, если $P_{ош} = 10^{-6}$ .

40. Порождающая матрица кода имеет вид

$$G = \begin{vmatrix} 1\,0\,0\,0\,0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,0\,0\,1\,1\,1\,0\,1 \\ 0\,0\,1\,0\,0\,1\,1\,0\,1\,0 \\ 0\,0\,0\,1\,0\,1\,1\,1\,1\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0\,1 \end{vmatrix}.$$

Найти проверочную матрицу кода; изобразить схемы кодирующего и декодирующего устройств; найти $d$ кода; определить возможности кода по обнаружению и исправлению ошибок.

41. Проверочная матрица имеет вид

$$H = \begin{vmatrix} 1\,1\,1\,1\,1\,0\,0\,0 \\ 1\,0\,1\,1\,0\,1\,0\,0 \\ 1\,1\,0\,1\,0\,0\,1\,0 \\ 1\,1\,1\,0\,0\,0\,0\,1 \end{vmatrix}.$$

Найти порождающую матрицу кода; изобразить схемы кодирующего и декодирующего устройств; найти $d_{\min}$ кода; определить возможности кода по обнаружению и исправлению ошибок.

42. С борта космического аппарата (КА) “Марс 2002” передается телевизионное изображение поверхности планеты. Размеры кадра изображения 512×512 элементов, каждый элемент квантуется на 128 уровней и кодируется с использованием сверточного (6,3)-кода. Мощность передатчика КА 100 Вт, коэффициент усиления антенны $\boldsymbol{G = 50}$.

Передача двоичных символов кодовых последовательностей осуществляется с использованием противоположных сигналов. Прием сигнала производится на антенну площадью $\boldsymbol{S = 100\ м^2}$, чувствительность приемника $\boldsymbol{N = 10^{-22}\ Вт/Гц}$.

За какое время может быть принят один кадр изображения, если прием производится посимвольно и отношение сигнал/шум по мощности на выходе приемника должно составить не менее 500? (Расстояние до КА $R = 50$ млн км).

Как изменится это время, если вместо помехоустойчивого кодирования использовать передачу элементов изображения ортогональными сигналами и осуществлять прием в целом?

43. В цифровой двоичной системе связи информация передается со скоростью 1 Мбит/с, при этом в среднем один раз в минуту в канале происходит ошибка.

Как изменится частота ошибок, если в канале использовать кодирование (7,3)-кодом Хемминга и для сохранения скорости передачи информации длительность передаваемых символов будет уменьшена в (7/3) раза? Определить величину выигрыша (проигрыша) по частоте ошибок за счет кодирования при средней вероятности ошибок в канале без кодирования $\boldsymbol{P_{ош} = 10^{-3} \ldots 10^{-6}}$.

44. Линейный блочный код задан порождающей матрицей $\boldsymbol{G}$ вида

$$\boldsymbol{G} = \begin{vmatrix} \mathbf{1\,0\,0\,1\,1\,0} \\ \mathbf{0\,1\,0\,0\,1\,1} \\ \mathbf{0\,0\,1\,1\,0\,1} \end{vmatrix}.$$

Изобразить схему кодера и синдромного декодера для этого кода. Составить таблицу декодирования с исправлением одиночных ошибок для декодера максимального правдоподобия.

45. Составить структурные схемы кодера и синдромного декодера для циклического (7,4)-кода, заданного порождающим полиномом $g(x) = 1 + x + x^3$.

Описать процесс кодирования и декодирования с исправлением одиночных ошибок.

46. Двоичный циклический код, заданный порождающим полиномом

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

позволяет исправлять пакеты ошибок длиной 2 (двойные ошибки в соседних символах).

Чему равна длина этого кода?

Найти минимальное кодовое расстояние данного кода.

Сконструировать систематический кодер для этого кода.

Сконструировать декодер, позволяющий исправлять пакеты по 2 ошибки.

47. Построить кодирующее и декодирующее устройства по схеме Меггитта для циклического (15,11)-кода Хемминга, имеющего порождающий многочлен вида

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

Сколько ошибок в принятой последовательности может обнаружить и исправить данный код?

# ПРИЛОЖЕНИЯ

## Приложение 1

Таблица многочленов сверточных кодов (*TablSvert*)

| **Номер Степени многочленов** **g(x)** мн-на $J$ - число проверок, $n_r$ -max[deg $g(x)$] | | **Номер Степени многочленов** **g(x)** мн-на $J$ - число проверок, $n_r$ -max[deg $g(x)$] | |
|---|---|---|---|
| № | $J$ $n_r$ | № | $J$ $n_r$ |
| { 1 : 2: 1 } | (0,1) | { 50: 4: 38} | (0,20,30,38) |
| { 2 : 2: 2 } | (0,2) | { 51: 4: 39} | (0,22,26,39) |
| { 3 : 2: 3 } | (0,3) | { 52: 4: 40} | (0,29,34,40) |
| { 4 : 2: 4 } | (0,4) | { 53: 4: 42} | (0,3,19,42) |
| { 5 : 3: 3 } | (0,2,3) | { 54: 4: 43} | (0,21,34,43) |
| { 6 : 3: 5 } | (0,3,5) | { 55: 4: 45} | (0,7,17,45) |
| { 7 : 3: 7 } | (0,1,7) | { 56: 4: 46} | (0,3,19,46) |
| { 8 : 3: 7 } | (0,3,7) | { 57: 4: 46} | (0,5,20,46) |
| { 9 : 3: 8 } | (0,1,8) | { 58: 4: 47} | (0,29,33,47) |
| { 10: 3: 8 } | (0,2,8) | { 59: 5: 11} | (0,2,7,10,11) |
| { 11: 3: 8 } | (0,3,8) | { 60: 5: 18} | (0,4,10,15,18) |
| { 12: 3: 8 } | (0,6,8) | { 61: 5: 22} | (0,2,9,21,22) |
| { 13: 3: 9 } | (0,2,9) | { 62: 5: 25} | (0,15,19,21,25) |
| { 14: 3: 10} | (0,4,10) | { 63: 5: 26} | (0,3,13,24,26) |
| { 15: 3: 10} | (0,1,10) | { 64: 5: 34} | (0,7,25,26,34) |
| { 16: 3: 11} | (0,5,11) | { 65: 5: 40} | (0,8,17,39,40) |
| { 17: 3: 11} | (0,6,11) | { 66: 5: 41} | (0,3,13,36,41) |
| { 18: 3: 12} | (0,1,12) | { 67: 5: 45} | (0,30,35,41,45) |
| { 19: 3: 12} | (0,11,12) | { 68: 5: 46} | (0,18,25,44,46) |
| { 20: 3: 12} | (0,3,9) | { 69: 5: 50} | (0,14,34,47,50) |
| { 21: 3: 14} | (0,10,14) | { 70: 5: 53} | (0,24,46,50,53) |
| { 22: 3: 15} | (0,1,15) | { 71: 5: 53} | (0,25,27,46,53) |
| { 23: 3: 15} | (0,2,15) | { 72: 5: 67} | (0,27,56,61,67) |
| { 24: 3: 17} | (0,4,17) | { 73: 5: 71} | (0,20,30,38,71) |
| { 25: 3: 17} | (0,7,17) | { 74: 5: 73} | (0,1,15,36,73) |
| { 26: 3: 18} | (0,4,18) | { 75: 5: 75} | (0,43,66,68,75) |
| { 27: 3: 18} | (0,10,18) | { 76: 5: 76} | (0,45,48,64,76) |
| { 28: 3: 19} | (0,3,19) | { 77: 5: 78} | (0,3,19,52,78) |
| { 29: 3: 20} | (0,15,20) | { 78: 5: 82} | (0,21,25,39,82) |
| { 30: 3: 22} | (0,13,22) | { 79: 5: 83} | (0,38,48,55,83) |
| { 31: 4: 6 } | (0,2,5,6) | { 80: 5: 84} | (0,5,20,47,84) |
| { 32: 4: 12} | (0,8,9,12) | { 81: 5: 85} | (0,11,12,62,85) |
| { 33: 4: 13} | (0,6,11,13) | { 82: 5: 88} | (0,2,8,32,88) |
| { 34: 4: 18} | (0,8,17,18) | { 83: 5: 89} | (0,36,67,76,89) |
| { 35: 4: 19} | (0,3,15,19) | { 84: 6: 17} | (0,2,7,13,16,17) |
| { 36: 4: 21} | (0,16,20,21) | { 85: 6: 38} | (0,8,27,28,32,38) |
| { 37: 4: 24} | (0,11,18,24) | { 86: 6: 39} | (0,14,21,23,36,39) |
| { 38: 4: 25} | (0,2,10,25) | { 87: 6: 50} | (0,15,22,23,40,50) |
| { 39: 4: 25} | (0,4,12,25) | { 88: 6: 58} | (0,3,5,14,34,58) |
| { 40: 4: 26} | (0,14,17,26) | { 89: 6: 61} | (0,12,16,42,48,61) |
| { 41: 4: 27} | (0,7,9,27) | { 90: 6: 71} | (0,5,26,51,55,71) |
| { 42: 4: 28} | (0,5,11,28) | { 91: 6: 72} | (0,6,7,41,60,72) |
| { 43: 4: 31} | (0,1,15,31) | { 92: 6: 77} | (0,10,32,47,49,77) |
| { 44: 4: 31} | (0,3,19,31) | { 93: 6: 82} | (0,8,11,24,44,82) |
| { 45: 4: 32} | (0,23,25,32) | { 94: 6: 96} | (0,46,54,80,83,96) |
| { 46: 4: 32} | (0,2,8,32) | { 95: 6:100} | (0,6,47,64,65,100) |
| { 47: 4: 32} | (0,10,29,32) | { 96: 6:101} | (0,38,40,62,71,101) |
| { 48: 4: 36} | (0,1,15,36) | { 97: 6:103} | (0,10,14,21,88,103) |
| { 49: 4: 37} | (0,25,36,37) | { 98: 6:104} | (0,5,28,48,60,104) |
| | | { 99: 6:118} | (0,42,87,90,106,118) |

## Приложение 2

Таблица многочленов сверточных кодов (*TablSvert*)

| **Номер** мн-на № | $J$ | $n_r$ | **Степени многочленов g(x)** $J$ - число проверок, $n_r$ -max[deg $g(x)$] |
|---|---|---|---|
| { 1 : | 2: | 1 } | (0,1) |
| { 2 : | 2: | 2 } | (0,2) |
| { 3 : | 2: | 3 } | (0,3) |
| { 4 : | 2: | 4 } | (0,4) |
| { 5 : | 3: | 3 } | (0,2,3) |
| { 6 : | 3: | 5 } | (0,3,5) |
| { 7 : | 3: | 7 } | (0,1,7) |
| { 8 : | 3: | 7 } | (0,3,7) |
| { 9 : | 3: | 8 } | (0,1,8) |
| { 10: | 3: | 8 } | (0,2,8) |
| { 11: | 3: | 8 } | (0,3,8) |
| { 12: | 3: | 8 } | (0,6,8) |
| { 13: | 3: | 9 } | (0,2,9) |
| { 14: | 3: | 10} | (0,4,10) |
| { 15: | 3: | 10} | (0,1,10) |
| { 16: | 3: | 11} | (0,5,11) |
| { 17: | 3: | 11} | (0,6,11) |
| { 18: | 3: | 12} | (0,1,12) |
| { 19: | 3: | 12} | (0,11,12) |
| { 20: | 3: | 12} | (0,3,9) |
| { 21: | 3: | 14} | (0,10,14) |
| { 22: | 3: | 15} | (0,1,15) |
| { 23: | 3: | 15} | (0,2,15) |
| { 24: | 3: | 17} | (0,4,17) |
| { 25: | 3: | 17} | (0,7,17) |
| { 26: | 3: | 18} | (0,4,18) |
| { 27: | 3: | 18} | (0,10,18) |
| { 28: | 3: | 19} | (0,3,19) |
| { 29: | 3: | 20} | (0,15,20) |
| { 30: | 3: | 22} | (0,13,22) |
| { 31: | 4: | 6 } | (0,2,5,6) |
| { 32: | 4: | 12} | (0,8,9,12) |
| { 33: | 4: | 13} | (0,6,11,13) |
| { 34: | 4: | 18} | (0,8,17,18) |
| { 35: | 4: | 19} | (0,3,15,19) |
| { 36: | 4: | 21} | (0,16,20,21) |
| { 37: | 4: | 24} | (0,11,18,24) |
| { 38: | 4: | 25} | (0,2,10,25) |
| { 39: | 4: | 25} | (0,4,12,25) |
| { 40: | 4: | 26} | (0,14,17,26) |
| { 41: | 4: | 27} | (0,7,9,27) |
| { 42: | 4: | 28} | (0,5,11,28) |
| { 43: | 4: | 31} | (0,1,15,31) |
| { 44: | 4: | 31} | (0,3,19,31) |
| { 45: | 4: | 32} | (0,23,25,32) |
| { 46: | 4: | 32} | (0,2,8,32) |
| { 47: | 4: | 32} | (0,10,29,32) |
| { 48: | 4: | 36} | (0,1,15,36) |
| { 49: | 4: | 37} | (0,25,36,37) |
| { 50: | 4: | 38} | (0,20,30,38) |
| { 51: | 4: | 39} | (0,22,26,39) |

```
{ 52: 4: 40}     (0,29,34,40)
{ 53: 4: 42}     (0,3,19,42)
{ 54: 4: 43}     (0,21,34,43)
{ 55: 4: 45}     (0,7,17,45)
{ 56: 4: 46}     (0,3,19,46)
{ 57: 4: 46}     (0,5,20,46)
{ 58: 4: 47}     (0,29,33,47)
{ 59: 5: 11}     (0,2,7,10,11)
{ 60: 5: 18}     (0,4,10,15,18)
{ 61: 5: 22}     (0,2,9,21,22)
{ 62: 5: 25}     (0,15,19,21,25)
{ 63: 5: 26}     (0,3,13,24,26)
{ 64: 5: 34}     (0,7,25,26,34)
{ 65: 5: 40}     (0,8,17,39,40)
{ 66: 5: 41}     (0,3,13,36,41)
{ 67: 5: 45}     (0,30,35,41,45)
{ 68: 5: 46}     (0,18,25,44,46)
{ 69: 5: 50}     (0,14,34,47,50)
{ 70: 5: 53}     (0,24,46,50,53)
{ 71: 5: 53}     (0,25,27,46,53)
{ 72: 5: 67}     (0,27,56,61,67)
{ 73: 5: 71}     (0,20,30,38,71)
{ 74: 5: 73}     (0,1,15,36,73)
{ 75: 5: 75}     (0,43,66,68,75)
{ 76: 5: 76}     (0,45,48,64,76)
{ 77: 5: 78}     (0,3,19,52,78)
{ 78: 5: 82}     (0,21,25,39,82)
{ 79: 5: 83}     (0,38,48,55,83)
{ 80: 5: 84}     (0,5,20,47,84)
{ 81: 5: 85}     (0,11,12,62,85)
{ 82: 5: 88}     (0,2,8,32,88)
{ 83: 5: 89}     (0,36,67,76,89)
{ 84: 6: 17}     (0,2,7,13,16,17)
{ 85: 6: 38}     (0,8,27,28,32,38)
{ 86: 6: 39}     (0,14,21,23,36,39)
{ 87: 6: 50}     (0,15,22,23,40,50)
{ 88: 6: 58}     (0,3,5,14,34,58)
{ 89: 6: 61}     (0,12,16,42,48,61)
{ 90: 6: 71}     (0,5,26,51,55,71)
{ 91: 6: 72}     (0,6,7,41,60,72)
{ 92: 6: 77}     (0,10,32,47,49,77)
{ 93: 6: 82}     (0,8,11,24,44,82)
{ 94: 6: 96}     (0,46,54,80,83,96)
{ 95: 6:100}     (0,6,47,64,65,100)
{ 96: 6:101}     (0,38,40,62,71,101)
{ 97: 6:103}     (0,10,14,21,88,103)
{ 98: 6:104}     (0,5,28,48,60,104)
{ 99: 6:118}     (0,42,87,90,106,118)
{100: 6:120}     (0,1,15,36,73,120)
{101: 6:122}     (0,55,82,111,116,122)
{102: 6:123}     (0,20,30,38,71,123)
{103: 6:124}     (0,49,92,115,117,124)
{104: 6:125}     (0,62,86,98,102,125)
{105: 6:126}     (0,21,25,39,82,126)
{106: 6:130}     (0,41,77,108,117,130)
{107: 6:131}     (0,11,12,62,85,131)
{108: 6:141}     (0,58,96,106,113,141)
{109: 6:142}     (0,2,8,32,88,142)
{110: 6:144}     (0,5,20,47,84,144)
{111: 6:146}     (0,3,19,52,78,146)
```

```
{112: 7: 25}     (0,2,7,15,21,24,25)
{113: 7: 46}     (0,1,28,31,44,46)
{114: 7: 54}     (0,7,17,29,40,49,54)
{115: 7: 69}     (0,12,17,45,48,68,69)
{116: 7: 90}     (0,6,10,32,40,47,90)
{117: 7: 91}     (0,13,27,29,38,73,91)
{118: 7:106}     (0,4,39,68,80,90,106)
{119: 7:110}     (0,14,60,61,66,91,110)
{120: 7:116}     (0,13,34,37,45,107,116)
{121: 7:122}     (0,2,17,59,95,115,122)
{122: 8: 35}     (0,7,10,16,18,30,31,35)
{123: 8: 77}     (0,27,39,47,61,64,70,77)
{124: 8: 78}     (0,19,21,45,63,73,74,78)
{125: 8:100}     (0,28,35,47,58,71,80,100)
{126: 8:124}     (0,16,31,34,48,75,85,124)
{127: 8:127}     (0,21,25,26,81,87,89,127)
{128: 8:141}     (0,19,59,68,85,88,103,141)
{129: 8:162}     (0,39,87,117,138,148,154,162)
{130: 8:172}     (0,2,13,25,96,118,168,172)
{131: 8:178}     (0,7,65,70,97,98,144,178)
{132: 9: 45}     (0,3,9,16,20,21,35,43,45)
{133: 9: 97}     (0,16,37,44,47,70,82,83,97)
{134: 9:102}     (0,22,24,30,41,73,93,98,102)
{135: 9:156}     (0,26,58,102,109,125,150,155,156)
{136: 9:160}     (0,12,29,39,72,91,146,157,160)
{137: 9:164}     (0,38,56,78,80,115,143,151,164)
{138:10: 55}     (0,2,14,21,29,32,45,49,54,55)
{139:10:116}     (0,23,39,57,60,74,101,103,112,116)
{140:10:130}     (0,1,6,25,32,72,100,108,120,130)
{141:10:152}     (0,8,38,48,59,82,111,146,150,152)
{142:10:199}     (0,12,25,28,78,83,100,109,145,199)
{143:10:202}     (0,7,27,76,113,137,155,156,170,202)
{144:11: 72}     (0,3,14,16,20,41,48,53,63,71,72)
{145:11:167}     (0,5,12,53,72,81,135,136,146,159,167)
{146:11:168}     (0,17,46,50,52,66,88,125,150,165,168)
{147:12: 85}     (0,2,6,24,29,40,43,55,68,75,76,85)
{148:12:193}     (0,26,34,47,57,58,112,121,140,181,188,193)
{149:12:195}     (0,17,46,50,52,66,88,125,150,165,168,195)
{150:13:114}     (0,23,28,29,37,53,63,75,94,96,107,111,114)
{151:13:240}     (0,16,64,104,122,124,146,159,167,171,178,237,240)
{152:13:250}     (0,30,39,53,68,129,139,165,170,216,222,249,250)
{153:14:127}     (0,5,28,38,41,49,50,68,75,92,107,121,123,127)
{154:14:279}     (0,2,7,42,45,117,163,185,195,216,229,246,255,279)
{155:14:288}     (0,8,12,27,28,64,113,131,154,160,208,219,233,288)
{156:15:155}     (0,4,5,15,33,57,59,78,105,117,125,139,142,148,155)
{157:15:334}     (0,45,59,86,103,179,202,230,245,263,267,279,298,309,334)
{158:15:336}     (0,13,60,70,108,110,162,182,183,188,191,273,297,329,336)
{159:16:179}     (0,6,19,40,58,67,78,83,109,132,133,162,165,169,177,179)
{160:17:201}     (0,18,24,46,50,67,103,112,115,126,128,159,166,167,186,196,201)
{161:18:216}     (0,2,10,22,53,56,82,83,89,98,130,148,153,167,188,192,205,216)
{162:19:246}     (0,1,6,25,32,72,100,108,120,130,153,169,187,190,204,231,233,
                                                                    242,246)
{163:20:283}     (0,24,30,43,55,71,75,89,104,125,127,162,167,189,206,215,272,
                                                                275,282,283)
{164:21:333}     (0,4,23,37,40,48,68,78,138,147,154,189,204,238,250,251,256,
                                                            277,309,331,333)
{165:22:359}     (0,3,16,45,50,51,65,104,125,142,182,206,210,218,228,237,289,
                                                        300,326,333,356,358)
{166:23:372}     (0,6,22,24,43,56,95,126,137,146,172,173,201,213,258,273,281,
                                                    306,311,355,365,369,372)
{167:24:425}     (0,22,41,57,72,93,99,139,147,173,217,220,234,273,283,285,296,
                                                303,328,387,388,392,416,425)
{168: 2: 5}      (0,5)
{169: 2: 6}      (0,6)
{170: 2: 7}      (0,7)
```

Библиографический список

1. *Питерсон, У.* Коды, исправляющие ошибки : пер. с англ. / У. Питерсон, Э. Уэлдон. – М. : Мир, 2006. – 594 с.

2. *Блейхут, Р.* Теория и практика кодов, контролирующих ошибки : пер. с англ. / Р. Блейхут ; под ред. К. Ш. Зигангирова. – М. : Мир, 2008. – 576 с.

3. Помехоустойчивость и эффективность систем передачи информации/ А. Г. Зюко [и др.] ; под ред. А. Г. Зюко. – М. : Радио и связь, 2007. – 272 с.

4. Микропроцессорные кодеры и декодеры / В. М. Муттер, Г. А. Петров [и др.]. – М. : Радио и связь, 2010. – 184 с.

5. *Макаров, А. А.* Автоматизация проектирования систем передачи данных : учеб. пособие / А. А. Макаров, В. И. Ковязин. ОЭИС. – Одесса, 2011. – 85 с.

6. *Артемова, О. А.* Исследование эффективности помехоустойчивого кодирования при передаче данных сложными сигналами в телефонных каналах : автореф. дис. ... канд. техн. наук. – Новосибирск, 2007. – 19 с.

7. Теория передачи сигналов : учеб. для вузов / А. Г. Зюко [и др.]. – М. : Радио и связь, 2005. – 304 с.

8. *Кларк, Дж.* Кодирование с исправлением ошибок в системах цифровой связи : пер. с англ. / Дж. Кларк, Дж. Кейн ; под ред. Б.С. Цыбакова. – М. : Радио и связь, 2009. – 392 с.

9. *Злотник, Б. М.* Помехоустойчивые коды в системах связи / Б. М. Злотник. – М. : Радио и связь, 2009. – 232 с.

10. *Макаров, А. А.* Методы повышения помехоустойчивости систем связи : учеб. пособие / А. А. Макаров ; Новосиб. электротехн. ин-т связи. – Новосибирск, 2012. – 58 с.

11. *Месси, Дж.* Пороговое декодирование / Дж. Месси. – М. : Мир, 2009. – 207 с.

12. *Robinson, J. P., Bernstein A. J.* A class of binary recurrent codes with limited error propagation. IEEE Trans. Inf. Theory, IT-13, January, 2010. – С. 106 – 113.

13. *Klieber, E. J.* Some difference triangles for constructing seef orthogonal codes. IEEE, Trans. Inf. Theory, IT-16, March, 2005. – С. 237 – 238.
14. *Брауде-Золотарев, Ю. Н.* Оптимизация порогового декодирования / Ю. Н. Брауде-Золотарев, В. В. Золотарев //Тр. НИИР, 2009. – № 1. – С. 40 – 45.
15. Пат. № 208153 Российская Федерация, H03 M13/12, 2004. Устройство для порогового декодирования сверточных кодов. О. А. Артемова, А. А. Макаров.
16. *Макаров, А. А.* Сверточные коды для итерационного декодирования : материалы междунар. науч.-практ. конф. «Информатика и проблемы телекоммуникаций» / А. А. Макаров. – Новосибирск, 2009. – С.156 – 157.
17. *Он же.* Итерационные декодеры свёрточных кодов : материалы междунар. семинара «Перспективы развития современных средств и систем телекоммуникаций» / А. А. Макаров, О. А. Артемова. – Хабаровск, 2010. – С. 141 – 149.
18. *Бронштейн, И. Н.* Справочник по математике для инженеров и учащихся вузов / И. Н. Бронштейн, К. А. Семендяев. – М. : Наука, 2012. – 544 с.

ОГЛАВЛЕНИЕ

*Учебное издание*

ЖУРАВЛЕВ Владимир Георгиевич
КУРАНОВА Наталья Юрьевна
ЕВСЕЕВА Юлия Юрьевна

ПОМЕХОУСТОЙЧИВЫЕ КОДЫ

Учебное пособие

Подписано в печать 06.11.13.
Формат 60x84/16. Усл. печ. л. 5,58. Тираж 50 экз.
Заказ
Издательство
Владимирского государственного университета
имени Александра Григорьевича и Николая Григорьевича Столетовых.
600000, Владимир, ул. Горького, 87.