МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра вычислительной техники

УТВЕРЖДАЮ
Проректор по учебной работе
_____О.Г.Локтионова
«5» 10 _____2023 г

Разработка кодеков кодов Хемминга и Рида-Соломона

Методические рекомендации по выполнению курсового проекта
для студентов, обучающихся по направлению подготовки09.04.01
«Информатика и вычислительная техника»

Курск 2023

УДК 621.391

Составитель:С.И.Егоров

Рецензент
Доктор технический наук, профессор кафедры ВТ Юго-Западного государственного университета *В.С.Титов*

Разработка кодеков кодов Хемминга и Рида-Соломона: методические рекомендации по выполнению курсового проекта / Юго-Зап. гос. ун-т;сост. : С.И. Егоров. - Курск, 2023. - 29 с.: ил. 4, табл. 1, прилож. 2. Библиогр.: с. 25.

Излагаются методические рекомендации по выполнению курсового проекта. Содержанием курсового проекта является разработкакодеков кодов Хемминга и Рида-Соломона.

Предназначены для студентов, обучающихся по направлению подготовки 09.04.01 «Информатика и вычислительная техника» дневной и заочной форм обучения.

Текст печатается в авторской редакции

Подписано в печать Формат 60x84 1/16.

Усл. печ. л. 0,9 Уч.-изд. л. 0,8 Тираж 50 экз. Заказ Бесплатно.

Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Содержание.

1. Введение.

Целью курсового проектирования по дисциплине "Технические средства защиты и сжатия информации" является:

– формирование навыковпроектирование кодеков наиболее популярных блочных помехоустойчивых кодов: а именно кодов Хемминга и Рида-Соломона;

– обобщение, закрепление и углубление знаний по дисциплине "Технические средства защиты и сжатия информации";

– формирование навыков разработки и оформления текстовой и графической технической документации.

Содержанием курсового проекта является проектирование кодеков кодов Хемминга и Рида-Соломона. При этом необходимопостроить проверочную и порождающую матрицы кода Хемминга, разработать кодер и декодер кода Хемминга, проверить на примере работоспособность кодека Хемминга. Для кода Рида-Соломона необходимо получить порождающий многочлен кода, разработать кодер и декодер, декодировать слово с ошибками.

2. Краткие теоретические сведения.

2.1. Основные понятия помехоустойчивого кодирования информации.

Реальные системы передачи данных несовершенны. Возможность возникновения ошибок (вероятность ошибок) должна учитываться при передаче и хранении информации. В первую очередь это относится к
– хранению информации на носителях с высокой плотностью записи: магнитные носители, CD, DVD;
– передаче данных при ограниченной мощности сигнала: спутниковая и мобильная связь;
– передаче информации по сильно зашумленным каналам: мобильная связь, высокоскоростные проводные линии связи;
– каналам связи с повышенными требованиями к надежности информации: вычислительные сети, линии передачи со сжатием данных.

Во всех перечисленных случаях используются помехоустойчивые коды, т.е. коды, исправляющие ошибки или ErrorCorrectionCodes (**ECC**).

Структура канонической цифровой системы связи приведена на рис. 1. Кодер вносит в передаваемые данные избыточность, необходимую для исправления ошибок декодером. Тем самым обеспечивается надежная передача данных через канал с ошибками.

Надежность передачи характеризуется средней долей ошибочных бит в данных или BitErrorRate (**BER**), которая определяется, как средняя вероятность ошибки одного бита передаваемой информации.

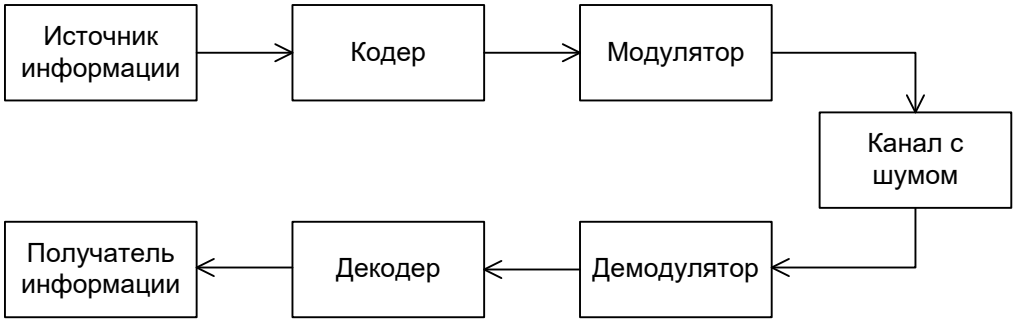

Рис. 1. Каноническая цифровая система связи.

Задача помехоустойчивого кодирования состоит в таком добавлении к информационным символам дополнительных символов, чтобы в приемнике искажения могли быть найдены и исправлены.

Определение. Двоичным кодом мощности Ми длины n называется множество из Мдвоичных словдлины n, называемыми кодовыми словами. Обычно $M = 2^k$, где k – некоторое целое число.Такой код называется (n, k)-кодом.

Пример. М =4, n=5, (n, k)=(5,2),

$$C = \begin{Bmatrix} 00000 \\ 01101 \\ 10011 \\ 11110 \end{Bmatrix}.$$

Данным кодом можно закодировать 2 бита, используя любое взаимное однозначное соответствие для кодирования, например:

$$\begin{matrix} 00 \leftrightarrow 00000 \\ 01 \leftrightarrow 01101 \\ 10 \leftrightarrow 10011 \\ 11 \leftrightarrow 11110 \end{matrix}.$$

Если получено одно из четырех 5-битовых слов, то полагаем, что соответствующие ему два бита являются правильной информацией. Если произошла ошибка при передаче, то получим 5-битовое слово, отличающееся от кодовых слов. Тогда попытаемся найти наиболее вероятное переданное слово и возьмем его в качестве оценки исходных двух битов информации. Например, если мы приняли 10111, то полагаем, что передавалось 10011, и, следовательно, информационное слово равнялось 10.

В общем случае блоковые коды определяются над произвольным конечным алфавитом из q символов {0, 1, 2, …, q-1}.

Определение. Блоковый код мощности M над алфавитом из q символов определяется как множество из Mq-ичных последовательностей длины n, называемых кодовыми словами. Обычно M = q^k для некоторого целого k.

Определение.Скорость блокового кода определяется равенством R = k / n. Скорость кода всегда меньше единицы: R< 1.

О блоковом коде судят по трем параметрам: длине блока n, информационной длине kи минимальному расстоянию d*. Минимальное расстояние является мерой различия двух наиболее похожих кодовых слов. Минимальное расстояние вводится двумя определениями.

Определение. Расстоянием по Хеммингу между двумя q-ичными последовательностями **x** и **y** длины n называется число позиций, в которых они различны. Это расстояние обозначается через d (**x**, **y**).

Например, возьмем **x** = 10101 и **y** = 01100; тогда имеем d (**x**, **y**) = d (10101, 01100) = 3. В качестве другого примера возьмем **x** = 20102 и **y** = 21103; тогда d (**x**, **y**) = d (20102, 21103) = 2.

Определение. Пусть C = {c_i, i = 0, …, M – 1} – некоторый код. Тогда минимальное расстояние кода C равно наименьшему из всех расстояний по Хеммингу между различными парами кодовых слов, т.е.

$$d^* = \min_{\substack{c_i, c_j \in C \\ i \neq j}} d(c_i, c_j).$$

(n, k)-код с минимальным расстоянием d* называется также (n, k, d*)-кодом.

Для кода, приведенного в примере в самом начале:

d(**x**, **y**)	00000	01101	10011	11110
00000	–	3	3	4
01101	3	–	4	3
10011	3	4	–	3
11110	4	3	3	–

Следовательно, для этого кода $d^*= 3$.

Если в канале произошло *t*ошибок и если расстояние от принятого слова до каждого другого кодового слова больше *t*, то декодер исправит эти ошибки, приняв ближайшее к принятому кодовое слово в качестве действительно переданного. Это будет всегда так, если $d^* \geq 2t + 1$. Для кода, приведенного в качестве примера, $d^* = 3$, поэтому данный код может исправлять одну ошибку $t = 1$.Максимальное число гарантировано исправимых ошибок на кодовое слово определяется как $t = \lfloor (d^* - 1)/2 \rfloor$.

Отметим, что блоковый код с минимальным расстоянием d^*гарантируетобнаружение всех конфигураций ошибок, содержащих $d^* - 1$ или меньшее число ошибочных бит.

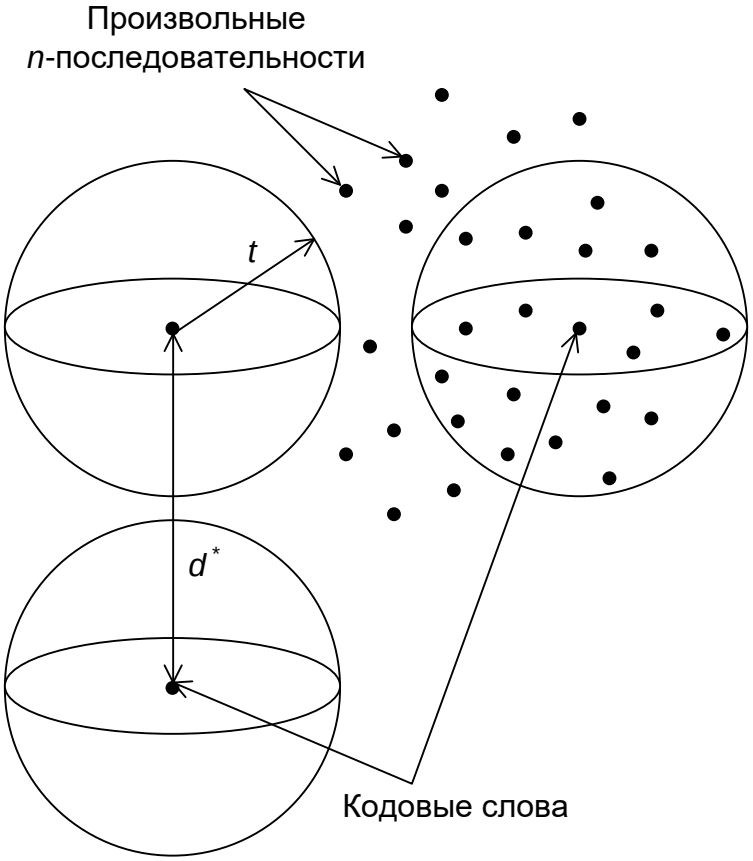

Рис. 2. Сферы декодирования.

Геометрическая иллюстрация приведена на рис. 2. В пространстве всех *q*-ичныхn-последовательностей выбрано некоторое подмножество

n-последовательностей, объявленных кодовыми словами. Если d^* – минимальное расстояние кода, а t – наибольшее целое число, удовлетворяющее условию $d^* \geq 2t + 1$, то вокруг каждого кодового слова можно описать непересекающиеся сферы радиуса t. Принятые слова, лежащие внутри сфер, декодируются как кодовое слово, являющееся центром соответствующей сферы. Если произошло не более tошибок, то принятое слово всегда лежит внутри соответствующей сферы и декодируется правильно.

Определение. Совершенный код есть код, для которого сферы некоторого одинакового радиуса вокруг кодовых слов (сферы декодирования), не пересекаясь, покрывают все пространство. Для совершенного кода $d^* = 2t + 1$. Примером совершенного кода является код Хемминга.

Определение. Коды, в которых информационное слово может быть непосредственно выделено из соответствующего ему кодового слова, называются систематическими (разделимыми).

Различают декодеры с полным и неполным декодированием.

Определение. Неполный декодер декодирует только те принятые слова, которые лежат внутри сфер декодирования, описанных вокруг кодовых слов. Остальные принятые слова, содержащие более допустимого числа ошибок, декодер объявляет нераспознаваемыми. Такие конфигурации ошибок при неполном декодировании называются неисправляемыми. Большинство используемых декодеров являются неполными декодерами.

Определение. Полный декодер декодирует каждое принятое слово в ближайшее кодовое слово. Полный декодер используется в тех случаях, когда лучше угадывать сообщение, чем вообще не иметь никакой его оценки.

Пусть в канал передачи данных посылается кодовое слово $\mathbf{u_1}$. В результате искажения передаваемого слова в канале в общем случае (неполный декодер) могут иметь место четыре варианта приема и декодирования (рис. 3.).

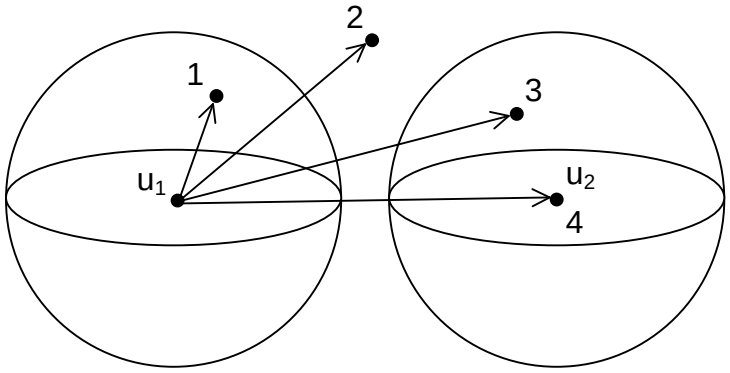

Рис. 3. Четыре варианта приема и декодированияв случае искажений.

1. Вектор ошибки $\mathbf{e_1}$отображает переданный вектор $\mathbf{u_1}$в точку, принадлежащую сфере декодирования $\mathbf{u_1}$. Декодер выдает потребителю переданное слово$\mathbf{u_1}$, исправляя при этом возникшие в канале ошибки.

2. Вектор ошибки $\mathbf{e_2}$ отображает переданный вектор$\mathbf{u_1}$в точку, не принадлежащую ни одной сфере декодирования, т.е. промежуточную между сферами декодирования область. Декодер распознает данную ситуацию и выдает потребителю сигнал отказа от декодирования, сигнализирующий об обнаружении неисправляемой конфигурации ошибок.

3. Вектор ошибки $\mathbf{e_3}$ отображает переданное кодовоеслово$\mathbf{u_1}$в точку, принадлежащую сфере декодирования кодового слова$\mathbf{u_2}$. Декодер выдает потребителю ошибочное кодовое слово $\mathbf{u_2}$. Таким образом, имеет место ошибочное декодирование.

4. Вектор ошибки $\mathbf{e_4}$ отображает переданное кодовоеслово$\mathbf{u_1}$в ошибочное кодовое слово $\mathbf{u_2}$. Декодер выдает потребителю ошибочное кодовое слово $\mathbf{u_2}$. Таким образом, имеет место необнаруживаемая ошибка.

Эффект от применения помехоустойчивого кодирования в системе передачи данных заключается в снижении вероятности ошибки на выходе приемника информации по сравнению с системой передачи данных, не использующей кодирование с коррекцией ошибок. Повышение достоверности данных на выходе системы при этом требует помимо введения новых компонентов (кодера и декодера) также увеличения скорости передачи (например, за счет увеличения полосы пропускания), поскольку кодирование с коррекцией ошибок избыточно.

2.2. Линейные коды.

Линейные коды составляют небольшую часть всех блоковых кодов. Однако, за редкими исключениями, именно линейные коды являются блоковыми кодами, имеющими практическое значение. Линейные коды также называются групповыми кодами или кодами с обобщенными проверками на четность.

Будем предполагать, что каждое кодовое слово линейного кода разбито на две части. Первая часть, состоящая из k символов, всегда совпадает с передаваемой информационной последовательностью. Каждый из n-k символов второй части вычисляется как линейная комбинация фиксированного подмножества информационных символов. Поэтому эти символы называются символами обобщенных проверок на четность или просто символами четности (paritysymbols).Как было сказано раннее, коды такого типа, в которых информационные символы при кодировании не изменяются, называются систематическими. Можно показать, что любой линейный код можно сделать систематическим на некотором множестве изk позиций, выбрав подходящее соответствие между входными последовательностями и кодовыми словами.

Линейные, или групповые, коды обладают свойством замкнутости, заключающего в том, что сумма любых двух кодовых слов также является кодовым словом.

Определение.Весω**(c)** кодового слова определяется как число ненулевых элементов этого слова. Легко видеть, что если рассмотреть посимвольную сумму по модулю 2 двух кодовых слов, то ее ненулевые символы соответствуют несовпадающим символам двух кодовых слов. Поэтому для любых двух кодовых слов **a** и **b** имеемd(**a**, **b**)=ω(**a**+**b**).

Отсюда вытекает, что множество расстояний от фиксированного кодового слова до всех других кодовых слов совпадает с множеством всех весов этого кода. Другая формулировка этого свойства состоит в том, что расстояние между двумя кодовыми словами совпадает с расстоянием от нулевого кодового слова до некоторого кодового слова. Следовательно, минимальное расстояние линейного кода равно минимальному весу ненулевого кодового слова. Таким образом, при построении группового кода с хорошим набором расстояний нужно стремиться к тому, чтобы веса ненулевых кодовых слов были возможно большими.

Граница Синглтона: минимальное расстояние (минимальный вес) любого линейного (n, k)-кода удовлетворяет неравенству $d^* \leq 1 + n - k$. Граница Синглтона показывает, что для исправления tошибок код должен иметь не менее $2t$проверочных символов.

Определение. Любой код с минимальным расстоянием, удовлетворяющим равенству $d^* = 1 + n - k$ называется кодом с максимальным расстоянием.

Своим названием групповые коды обязаны тому, что множество кодовых слов вместе с нулевым словом, снабженное операцией посимвольного сложения по модулю 2, образует математическую структуру, называемую группой.

Основные свойства группы таковы:

1. Сумма двух элементов группы всегда лежит в группе (замкнутость).

2. Выполняется закон ассоциативности, так что (a+b)+c=a+(b+c).

3. Группа всегда содержит единичный элемент (нулевое слово).

4. Каждый элемент группы обладает обратным (в случае двоичного кода каждое слово совпадает со своим обратным), для которого a + (-a) = 0.

Проверочная матрица.

Вместо того, чтобы явно выписывать уравнения для проверок на четность, часто оказывается более удобным использовать матричные обозначения. Например, проверочная матрица H содержит ту же информацию о коде, что и выписанное в общем виде кодовое слово **a**.

Примердля (6,3)-кода: $a^T = (a_1 , a_2 , a_3 , a_1 + a_2 , a_2 + a_3 , a_1 + a_2 + a_3)$.

Проверочная матрица будет при этом иметь следующий вид:

$$H = \begin{bmatrix} 110100 \\ 011010 \\ 111001 \end{bmatrix}.$$

Каждый столбец матрицы H соответствует некоторому символу кодового слова: первые три столбца–информационным символам, а последние три– проверочным. Соотношение, выражаемое первой строкой, состоит в том, что четвертый символ является суммой a_1 и a_2. Аналогично вторая строка указывает, что пятый символ является суммой a_2 и a_3 и т. д. Эта матрица записана в канонической форме. Это значит, что первые k столбцов задают информационные символы, которые входят во все уравнения, в то время как последние n-k столбцов образуют единичную матрицу.Таким образом, проверочная матрица имеет размерность $(n\text{-}k) \times n$.

Последовательность **a** является кодовым словом в том, и только в том случае, еслиHa = 0.В этом уравнении матричное умножение выполняется обычным образом, с тем единственным исключением, что в случае двоичных кодов используется сложение по модулю 2.

Для каждого фиксированного кодового слова уравнениеHa = 0означает, что сумма некоторого подмножества столбцов матрицы равна 0.

Например, последовательность (100101) является кодовым словом (6,3)-кода из предыдущего примера. При матричном умножении ненулевые элементы этой последовательности “отсеивают” первый, четвертый и шестой столбцы H, т. е.

$$\begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}.$$

Поскольку аналогичное соотношение должно выполняться для любого кодового слова, можно заметить следующее. Если минимальный вес, т. е. кодовое расстояние кода, равно d^*, то должно существовать по крайней мере одно подмножество, состоящее изd^* столбцов матрицы H, сумма которых равна 0. С другой стороны, не может существовать ни одного подмножества из$d^* - 1$ или менее столбцов, сумма которых равна 0. Если рассматривать столбцы матрицы H как векторы, то можно сказать, что для кода с кодовым расстояниемd^* все подмножества из$d^* - 1$ столбцов H должны быть линейно независимы. Это утверждение составляет одну из фундаментальных теорем о групповых кодах. Оно позволяет находить кодовое расстояние группового кода, заданного матрицей H, а также строить матрицы H, приводящие к кодам с гарантированным кодовым расстоянием.

В общем случае строки проверочной матрицы Нявляются линейно независимыми. Любая линейная комбинация строк может служить проверочным уравнением, и любое множество из n-k полученных таким способом линейно независимых уравнений может быть использовано для

образования матрицы H. Следовательно, в результате таких преобразований проверочную матрицу произвольного линейного кода можно привести к канонической форме.

2.3. Коды Хемминга.

Поскольку для получения кодового расстояния 3 все подмножества из двух столбцов проверочной матрицы должны быть линейно независимыми, достаточно, чтобы все столбцы были различными и ненулевыми. Таким образом, для трех проверочных символов имеется семь различных ненулевых троек, для четырех — пятнадцать различных ненулевых четверок и т. д. Это дает семейство кодов Хемминга, впервые описанных Ричардом В. Хеммингом в 1950 г.

Коды Хемминга позволяют исправлять одну ошибку, т.е. для этих кодов $t = 1$ и $d^* = 3$. Для каждого m существует $(2^m - 1,\ 2^m - 1 - m)$-код Хемминга. При больших m скорость кода близка к 1, но доля общего числа битов, которые могут быть искажены, очень мала. Коды Хемминга обладают несколькими замечательными свойствами. Прежде всего они являются примерами немногих известных совершенных кодов. Т.е. эти коды идеально упакованы.

Пример проверочной матрицы для (7,4)-кода Хемминга:

$$H = \begin{bmatrix} 1110100 \\ 1101010 \\ 1011001 \end{bmatrix}.$$

Коды Хемминга с минимальным кодовым расстоянием 3 можно превратить в расширенные коды Хемминга с кодовым расстоянием 4, добавив дополнительный проверочный символ, равный сумме всех остальных символов.Коды Хемминга с проверкой на четность обладают следующими преимуществами:

– длина кодов увеличивается с $2^m - 1$до 2^m, что удобно с точки зрения хранения и передачи информации;

– минимальное расстояние расширенных кодов увеличивается на единицу, что дает возможность обнаруживать ошибки большей кратности.

Порождающая матрица.

Так как сумма по модулю 2 любых двух кодовых слов снова является кодовым словом, то несколько раз используя это свойство, получаем, что любая линейная комбинация кодовых слов также является кодовым словом. Поскольку информационные символы выбираются независимо, можно надеяться, что существуют кодовые слова, каждое из которых содержит ровно один символ‘1’в информационной части кодового слова. Тогда все 2^k кодовых слов можно будет получить как 2^k возможных линейных комбинаций этихk базисных векторов, образующих линейноевекторное пространство. Если сформировать из этих четырех векторов матрицу, то получим так называемую порождающую

матрицукода. В канонической форме она всегда состоит из единичной матрицы порядка $k \times k$, к которой присоединена $k\times (n\text{-}k)$– матрица проверочных символов. Проверочная часть порождающей матрицы получается из матрицы H (в канонической форме) транспонированием подматрицы, образованной первымиk столбцами.

Пример для (7,4)-кода Хемминга:

проверочная матрица $$H = \begin{bmatrix} 1110100 \\ 1101010 \\ 1011001 \end{bmatrix},$$

соответствующая порождающая матрица $$G = \begin{bmatrix} 1000111 \\ 0100110 \\ 0010101 \\ 0001011 \end{bmatrix}.$$

Если порождающая матрица имеет вид $G = [I_k, P]$, где P – $k \times (n - k)$ матрица проверочных символов, а I – единичная матрица, то проверочная матрица при этом имеет вид $H = [-P^T, I_{n-k}]$, так как $GH^T = 0$. Таким образом, по порождающей матрице, записанной в канонической форме, сразу можно получить проверочную матрицу и наоборот. Любая порождающая матрица может быть приведена к такому частному виду при помощи последовательности элементарных операций над строками и последующей перестановки столбцов. Такая порождающая матрица называется порождающей матрицей в систематическом виде.

Пример кодирования данных с помощью порождающей матрицы. Пусть $i = (1,1,0,1)$ – передаваемое информационное слово, тогда соответствующее кодовое слово будет равно:

$$c = i \cdot G = (1101) \cdot \begin{bmatrix} 1000111 \\ 0100110 \\ 0010101 \\ 0001011 \end{bmatrix} = (1101010).$$

Полученный вектор с является кодовым словом, так как

$$H \cdot c^T = \begin{bmatrix} 1110100 \\ 1101010 \\ 1011001 \end{bmatrix} \cdot (1101010)^T = (000)^T = 0.$$

Синдромное декодирование.

Таблицей стандартного расположения кода называется таблица, содержащая в самой первой строке все возможные кодовые слова, а в самом крайнем левом столбце возможные конфигурации ошибок. Это расположение обладает тем свойством, что элементi-го столбца равен сумме кодового слова, записанного в верхней строке этого столбца, и соответствующей комбинации ошибок, записанной в соответствующей строке нулевого столбца. Другими словами, каждая строка расположения получается прибавлением фиксированной комбинации к каждому

кодовому слову. В теории групп строки такого расположения называются классами смежности, а элемент, стоящий в самом левом столбце, называется лидером класса смежности. Каждый элемент класса смежности может быть выбран в качестве лидера. Другой выбор лидера приведет лишь к изменению порядка элементов внутри класса смежности. Обычно в качестве лидера выбирают, конечно, элемент с наименьшим весом.

Пример таблицы стандартного расположения для (6,3)-кода, исправляющего одиночные ошибки, рассмотренного ранее:

Таблица 1. Таблица стандартного расположения для (6,3)-кода

000000	100101	010111	110010	001011	101110	011100	111001
000001	100100	010110	110011	001010	101111	011101	111000
000010	100111	010101	110000	001001	101100	011110	111011
000100	100001	010011	110110	001111	101010	011000	111101
001000	101101	011111	111010	000011	100110	010100	110001
010000	110101	000111	100010	011011	111110	001100	101001
100000	000101	110111	010010	101011	001110	111100	011001
000101	100000	010010	110111	001110	101011	011001	111100

Один из методов построения декодера состоит в том, чтобы записать эту таблицу в постоянную память (ПЗУ). Принятое слово является адресом, а кодовое слово– информацией, хранящейся по этому адресу. Табличные методы используются для кодов короткой длины. Сложность такой схемы можно заметно уменьшить, если удастся найти простое правило, определяющее класс смежности, в котором лежит принятая последовательность длины n. Тогда будет нужна существенно меньшая таблица, в которой необходимо будет искать требуемый лидер класса смежности или комбинацию ошибок. Затем эта коминация ошибок будет складываться по модулю 2 с принятой последовательностью. Такой подход действительно оказывается возможным.

Предположим, что принятое слово **r** содержит комбинацию ошибок **e**. Тогда**r** можно записать в видег = c + e.Умножив принятое слово **r** на матрицу H, получим вектор **s**, называемый синдромом:

$$s = Hr = Hc + He = He.$$

Таким образом, проверочная матрица позволяет определить вектор, зависящий только от комбинации ошибок, тем самым, убрав некоторую ненужную информацию для декодирования из принятого слова. Любые два слова, имеющие одинаковый синдром, отличаются на кодовое слово. По построению стандартного расположения элементы одной и той же строки удовлетворяют этому условию, а элементы двух разных строк не могут отличаться друг от друга на кодовое слово. Поэтому каждая строка стандартного расположения однозначно характеризуется соответствующим синдромом. Таким образом, задача декодирования упрощается и исправление ошибок сводится к нахождению для полученного синдрома лидера соответствующего смежного класса,

поскольку этот лидер является разностью между принятым словом и кодовым словом, т.е. вектором ошибок **e**. Следовательно, исправление ошибки может быть выполнено путем вычитания из принятого слова лидера смежного класса. Имеет смысл рассматривать процедуру жесткого декодирования линейных кодов как метод решения уравнения $He = s$. Любой метод решения этого уравнения является методом декодирования.

В процессе декодирования на основе вычисления синдрома могут возникнуть следующие ситуации:

1. $s = 0$, $e = 0$ – безошибочная передача информации.
2. $s \neq 0$, $e \neq 0$ – ошибка будет обнаружена при декодировании, но при этом она может оказаться неисправляемой, либо декодирование может быть ошибочным (случаи 1,2,3).
3. $s = 0$, $e \neq 0$ – передача информации с необнаруживаемой ошибкой, так как сам вектор ошибки является кодовым словом, т.е. $e \in C$ (случай 4).

2.4. Коды Рида-Соломона

Помехоустойчивые коды Рида-Соломона широко используются для обеспечения надежной передачи и хранения данных в информационных системах. Они применяются: в спутниковых (например, система DVB) и оптоволоконных каналах передачи цифровой информации, в оптических накопителях информации (CD, DVD, BlueRay).

Коды Рида-Соломона (РС-коды) являются линейными циклическими кодами. РС-коды являются подмножеством кодов Боуза - Чоудхури - Хоквингема (БЧХ) над полем Галуа GF(q), длина которых n равна q-1. При этом поле символов кода совпадает с полем локаторов ошибок (рассматриваются только РС-коды, определенные над конечным полем характеристики 2, для которых число элементов в поле $q=2^m$).

Порождающий многочлен РС-кода с минимальным кодовым расстоянием d имеет следующий вид:

$$G(x) = (x - \alpha^b)(x - \alpha^{b+1})...(x - \alpha^{b+d-2}), \quad (1)$$

где α - примитивный элемент GF(q); b - целочисленная константа.

Размерность РС-кода (количество информационных символов) равна $k=n-\deg G(x)=n-d+1$. Избыточность кода $R=(n-k)/n$. В другой записи $d=n-k+1$, и коды РС являются разделимыми кодами с максимальным расстоянием, или МДР-кодами. МДР-код имеет максимально возможное расстояние между кодовыми словами для заданных n и k, и кодовые слова могут быть разделены на информационные и проверочные символы (т.е. МДР-код имеет систематический кодер). Максимальное число гарантированно исправляемых ошибок в кодовом слове $t = \lfloor (d-1)/2 \rfloor$.

Укороченный код Рида-Соломона получается из основного кода Рида-Соломона путем удаления символов кодового слова в старших

информационных разрядах. При этом минимальное кодовое расстояние не меняется.

Для повышения эффективности исправления пакетов ошибок применяют перемежение символов нескольких кодовых слов. Это особенно важно для исправления протяженных пакетов ошибок, вызванных сбоем тактовой синхронизации. Параметр λ определяет глубину перемежения, указывая число перемеженных кодовых слов (при λ = 1 перемежение отсутствует).

Символы кодовых слов, передаваемые по каналу, являются элементами поля Галуа GF(2^m) и представляются в виде двоичного разложения в базисе α^{m-1}, α^{m-2}, ... , α^2, α^1, α^0. Первый бит каждого символа соответствует коэффициенту при α^{m-1}.

В основе разработанных к настоящему времени алгебраических алгоритмов декодирования кодов Рида-Соломона лежат различные методы решения нелинейной системы уравнений над конечными полями вида:

$$\begin{aligned} S_b &= Y_1 X_1^b + Y_2 X_2^b + \ldots + Y_v X^b; \\ S_{b+1} &= Y_1 X_1^{b+1} + Y_2 X_2^{b+1} + \ldots + Y_v X^{b+1}; \\ S_{b+2t-1} &= Y_1 X_1^{b+2t-1} + Y_2 X_2^{b+2t-1} + \ldots + Y_v X_v^{b+2t-1}, \end{aligned} \qquad (2)$$

где t - максимальное количество ошибок, исправляемое декодером,
v - количество ошибочных символов в обрабатываемом кодовом слове,
S_j - компонента синдрома (получается в результате подстановки в многочлен принятого слова α^j),
X_i - локатор ошибочного символа,
Y_i - значение ошибки.

Эту систему нелинейных уравнений трудно решать непосредственно. Путем введения многочлена:

$$\Lambda(x) = \Lambda_v x^v + \Lambda_{v-1} x^{v-1} + \ldots + \Lambda_1 x + 1, \qquad (3)$$

известного под названием многочлена локаторов ошибок, корнями которого являются обратные к локаторам ошибок величины X'_i, систему (2) преобразуют в систему линейных уравнений:

$$\begin{aligned} \Lambda_1 S_v + \Lambda_2 S_{v-1} + \cdots + \Lambda_v S_1 &= -S_{v+1}; \\ \Lambda_1 S_{v+1} + \Lambda_2 S_v + \cdots + \Lambda_v S_2 &= -S_{v+2}; \end{aligned} \qquad (4)$$

Определив число ошибок в слове и решив систему (4), можно получить многочлен локаторов ошибок. Корни многочлена дадут локаторы

ошибок. Зная локаторы ошибок, значения ошибок можно найти из системы (2).

Таким образом, в процедуре декодирования можно выделить следующие этапы:

1) вычисление синдрома,
2) определение количества ошибок в принятом слове и получение многочлена локаторов ошибок,
3) вычисление локаторов ошибок,
4) нахождение значений ошибок,
5) исправление ошибок.

Рассмотрим методы выполнения отдельных этапов процедуры декодирования.

Вычисление синдрома

Нахождение компонент синдрома заключается в вычислении множества значений многочлена над конечным полем. Пусть $R(x)$ - многочлен принятого из канала слова:

$$R(x) = r_{n-1}x^{n-1} + r_{n-2}x^{n-2} + \cdots + r_1 x + r_0 \qquad (5)$$

Компоненты синдрома равны: , где α^i - корни порождающего многочлена кода, $i = \overline{b, b+d-2}$.

Чаще всего значения $R(x)$ вычисляют по схеме Горнера. При этом используют представление многочлена $R(x)$ в следующем виде:

$$R(x) = (\ldots (r_{n-1}x + r_{n-2})x + \cdots + r_1)x + r_0 \qquad (6)$$

Эту процедуру применительно к блоку перемеженных кодовых слов можно описать с помощью рекуррентной формулы:

$$S_{i,j}^{t+1} = S_{i,j}^{t} \cdot \alpha^{i} + r_{n-1-t,j}; S_{i,j}^{t} = 0, i = \overline{b, b+d-2}, j = \overline{0, \lambda-1}, t = \overline{0, n-1}, \qquad (7)$$

где - символ j-го кодового слова; $S_{i,j}^{t}$ - i-ая компонента j-го синдрома после обработки tпервых символов j-го кодового слова.

Получение многочлена локаторов ошибок.

Многочлен локаторов ошибок получается в результате определения количества ошибок v и решения системы (4). Количество ошибок в кодовом слове дает ранг матрицы компонент синдрома

Определение ранга матрицы $\overline{S_t}$ и решение системы (4) может быть выполнено традиционными способами, не использующими

структуру матрицы $\overline{S_t}$, ее симметрию. При этом число операций в полях растет какt^3.

На практике для небольших t нашло применение вычисление коэффициентов многочлена локаторов ошибок по формулам, непосредственно связывающими их со значениями компонент синдрома.

Для t>4,5 используют алгоритмы, использующие свойства симметрии матрицы компонент синдрома. При этом число операций в полях растет как t^2. В число таких алгоритмов входят: алгоритм Берлекэмпа, алгоритм Берлекэмпа-Месси, алгоритм Евклида.

Вычисление локаторов ошибок.

Вычисление локаторов ошибок сводится к решению уравнения в поле GF(2^m)

В случае v=1, уравнение $\lambda_1 x + \lambda_0 = 0$ линейно, $X_1' = 1/\lambda_1$ и локатор ошибки равен λ_1. Для решения уравнений любой степени можно применять процедуру Ченя, которая заключается в последовательной подстановке всех возможных значений величин обратных к локаторам в многочлен и проверке результата на нулевое значение.

Вычисление значения многочлена в точке $x=X'$ можно выполнить по схеме Горнера.

Нахождение корней полинома локаторов ошибок второй и третьей степени при программной реализации целесообразно осуществлять, приводя с помощью подстановок степенное уравнение к виду:

$^2 + x + B = 0$, (8)

или $^3 + B = 0$, (9)

или $^3 + x + B = 0$; (10)

с последующим нахождением корней полученных уравнений с помощью таблиц.

Нахождение значений ошибок.

После определения на предыдущем этапе локаторов ошибок нелинейная система уравнений (2) становится линейной относительно неизвестных Y_iи может решаться известными методами решения линейных систем. Более эффективная процедура предложена Форни.

Значения ошибок получаются из равенства:

$$Y_i = \frac{X_i^{-1}\Omega(X_i^{-1})}{\prod_{j \neq i}(1 - X_j X_i^{-1})} - \frac{\Omega(X_i^{-1})}{\Lambda'(X_i^{-1}) X_i^{b-1}}, \quad (11)$$

где $\Lambda'(x)$ - формальная производная многочлена $\Lambda(x)$;

$\Omega(x)$ - многочлен значений ошибок $\Omega(x)=S(x)\Lambda(x) \bmod x^{2t}$;

- многочлен синдрома.

Исправление ошибок

Исправление ошибок заключается в прибавлении значения ошибкиY_i к принятому из канала символу r_k. Причем $k = (2^m - 2) - \log_\alpha X_i$.

3. Техническое задание на проектирование.

Содержанием курсового проекта является проектирование кодеков наиболее популярных блочных помехоустойчивых кодов: а именно кодов Хемминга и Рида-Соломона. Каждый кодек содержит две части: и декодер. При этом необходимо разработать кодеркода Хемминга, декодер кода Хемминга, привести пример работы кодека. Необходимо также разработать кодеркода Рида-Соломона, декодер кода Рида-Соломона, привести пример работы декодера.

Индивидуальные варианты заданий, выдаваемые по указанию преподавателя, приведены в приложении.

3.1. Требования к кодеку кода Хемминга.

Необходимо разработать кодек для кода Хемминга. Длина кодового слова, количество информационных и проверочных символов должны соответствовать варианту, указанному преподавателем (приложение 1).

На вход кодера параллельно подаются информационные биты. На выходе кодере должны формироваться биты кодового слова. Задержка кодера – один такт.

На вход декодера из канала параллельно подаются кодовые биты. На выходе декодера должны формироваться информационные биты кодового слова с исправленной ошибкой. Задержка декодера – тоже один такт.

Дополнительно на выходе должны формироваться флаги: отсутствия ошибок в кодовом слове, исправления одной ошибки, наличия неисправимых ошибок.

3.2. Требования к кодеку кода Рида-Соломона.

Необходимо разработать кодек для кода Рида-Соломона. Длина кодового слова, количество информационных и проверочных символов должны соответствовать варианту, указанному преподавателем (приложение 2). Многочлен, порождающий конечное поле, также определяет преподаватель.

На вход кодера последовательно подаются информационные символы. На выходе кодере должны последовательно формироваться символы кодового слова. Задержка кодера – один такт.

На вход декодера из канала последовательно подаются символы кодового слова. На выходе декодера должны формироваться символы кодового слова с исправленными ошибками.

Вариант кодового слова с ошибками для проверки декодера указывает преподаватель (приложение 2).

4. Содержание этапов проектирования.

В курсовом проекте предлагается следующий состав и порядок выполнения этапов:

– расшифровка и анализ задания;

– разработка кодека кода Хемминга;

–разработка кодека кода Рида-Соломона;

– оформление проекта.

4.1. Расшифровка задания.

Задание на курсовой проект представляется в виде двух заданий:задания на разработку кодека кода Хемминга и задания на разработку кодека кода Рида-Соломона.

В задании для кодека кода Хемминга указывается, какой код Хемминга используется: обычный или расширенный. В задании также указаны: длина кодового слова, количество информационных и проверочных бит.

В задании для кодека кода Рида-Соломона указываются базовые параметры кода Рида-Соломона: длина кодового слова; количество информационных и проверочных символов; многочлен, задающий поле Галуа.

4.2. Разработка кодека кода Хемминга.

Рассмотрим разработку кодека на примере кода Хемминга с n = 10, k = 6, m = 4, где n–длина кодового слова, k — количество информационных бит, m — число проверочных бит.

1) Строится проверочная матрица H для данного кода Хемминга.

Для кода Хемминга все столбцы проверочной матрицы H должны быть различными и ненулевыми. Матрица H в канонической форме содержит в своей правой части единичную подматрицу размерности $m*m$. Эта подматрица, со столбцами веса 1, соответствует проверочным символам кода. В качестве столбцов матрицы, соответствующих информационным символам, могут быть выбраны любые двоичные векторазмерности mс весом большим 1. Например:

$$H = \begin{array}{c} \begin{matrix} c_9 & c_8 & c_7 & c_6 & c_5 & c_4 & c_3 & c_2 & c_1 & c_0 \end{matrix} \\ \left|\begin{matrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{matrix}\right| \\ \begin{matrix} i_5 & i_4 & i_3 & i_2 & i_1 & i_0 \end{matrix} \end{array} \begin{matrix} s_3 \\ s_2 \\ s_1 \\ s_0 \end{matrix} . \quad (12)$$

Расширенный код Хемминга получается из обычного путем добавления дополнительного проверочного бита, равного сумме по модулю

2 всех бит кодового слова базового кода. Проверочная матрица расширенного кода получается путем добавления дополнительной строки единиц в проверочную матрицу базового кода.

2) Строится порождающая матрица G для данного кода Хемминга

Порождающая матрица кода Хемминга в канонической форме имеет вид G = [I_k ,P], где P – $k \times (n - k)$ матрица проверочных символов, а I – единичная матрица, и H = [$-P^T$, I_{n-k}], так как $GH^T = 0$. Проверочная матрица получается из порождающей путем присоединения к единичной подматрице справа подматрицы P, получаемой транспонированием подматрицы четности матрицы H. Например:

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

3) Разрабатывается схема кодера Хемминга

Обозначим кодовое слово кода Хемминга:

$$C = (c_9, c_8, c_7, c_6, c_5, c_4, c_3, c_2, c_1, c_0),$$

где разряды $, c_3, c_2, c_1, c_0$ - проверочные.

Исходя из порождающей матрицы *G*запишем выражения для определения проверочных символовкодового слова как сумм различных подмножеств символов информационного слова:

$c_3 = i_5 + i_4$;

$c_2 = i_3 + i_2 + i_1$;

$c_1 = i_5 + i_3 + i_2 + i_0$;

$c_0 = i_5 + i_4 + i_3 + i_1 + i_0$.

Нетривиальная часть кодера реализует вычисления вышеприведенных выражений.

4) Разрабатывается схема декодера Хемминга

Декодер Хемминга реализует синдромное декодирование.

Сначала вычисляется синдром принятого из канала слова путем умножения принятого из канала слова R на транспонированную проверочную матрицу H. Эта операция реализуется путем вычисления следующих логических функций:

$s_3 = c_9 + c_8 + c_3$;

$s_2 = c_7 + c_6 + c_5 + c_2$;

$s_1 = c_9 + c_7 + c_6 + c_4 + c_2$;

$s_0 = c_9 + c_8 + c_7 + c_5 + c_4 + c_0$.

В случае одной ошибки в кодовом слове значение синдрома будет равняться столбцу матрицы H, соответствующему позиции ошибки. С помощью дешифратора, на который подается синдром, определяется позиция, в которой произошла ошибка. Бит в этой позиции инвертируется с помощью сумматора по модулю 2.

5) Разрабатывается тестовый пример работы кодека Хемминга

Символы информационных слов тестовой последовательности получаются следующим образом. Берется последовательность букв фамилии студента. Каждая буква заменяется на свой числовой эквивалент, представленный 5 битами в соответствии с расположением буквы в алфавите. Полученная битовая последовательность нарезается на *k*-разрядные информационные слова.

Информационные слова кодируются затем кодером Хемминга. Получается последовательность слов кода Хемминга.

В указанный преподавателем бит одного из кодовых слов вносится ошибка. Для этого слова вычисляется синдром и проверяется правильность исправления ошибки декодером.

4.3. Разработка кодека кода Рида-Соломона.

1) ***Разрабатывается кодер кода Рида-Соломона.***

Для кода Рида-Соломона с заданными параметрами вычисляется многочлен $G(x)$, порождающий код. Затем разрабатывается кодер Рида-Соломона в виде регистра сдвига с обратными связями, в которых осуществляется умножение выходного сигнала на коэффициенты многочлена $G(x)$.

2) ***Разрабатывается вычислитель синдрома декодера кода Рида-Соломона.***

Компоненты синдрома равны: $S_i = R(\alpha^{b+i-1})$, где α^{b+i-1} - корни порождающего многочлена кода, $i = \overline{0, d-2}$, b = номер варианта по модулю $2^m - 1$.

Значения $R(x)$ вычисляются по схеме Горнера. При этом используют представление многочлена $R(x)$ в следующем виде:

$$R(x) = (\ldots (r_{n-1}x + r_{n-2})x + \cdots + r_1)x + r_0.$$

Необходимо разработать умножители на постоянные коэффициенты α^{b+i-1}.

3) ***Разрабатывается вычислитель локаторов (позиций) и значений ошибок.***

Вычисление локаторов ошибок сводится к нахождению корней уравнения в поле GF(2^m)

Для этого применяется процедура Ченя, которая заключается в последовательной подстановке всех возможных значений величин обратных к локаторам в многочлен и проверке результата на нулевое значение. При этом последовательные вычисления значений многочлена в точке $x=X'$ выполняются по схеме Горнера.

Одновременно с обнаружением ошибки осуществляется вычисление ее значения и исправление.

Значения ошибок получают по методу Форни:

$$Y_i = \frac{X_i^{-1}\Omega(X_i^{-1})}{\prod_{j\neq i}(1-X_j X_i^{-1})} - \frac{\Omega(X_i^{-1})}{\Lambda'(X_i^{-1})X_i^{b-1}},$$

где $\Lambda'(x)$ - формальная производная многочлена $\Lambda(x)$;

$\Omega(x)$ - многочлен значений ошибок $\Omega(x)=S(x)\Lambda(x) \bmod x^{2t}$;

- многочлен синдрома.

4) ***Выполняется проверка правильности работы разработанного декодера кода Рида-Соломона.***

Вариант кодового слова и ошибки для проверки декодера указывает преподаватель (приложение 2).

4.4. Оформление курсового проекта.

Курсовой проект оформляется согласно ГОСТ и ЕСКД. Объем пояснительной записки не может быть менее 20 страниц. К проекту должны прилагаться плакаты со следующим графическим материалом: 1) чертеж функциональной схемы кодека кода Хемминга, 2) чертеж функциональной схемы кодека кода Рида-Соломона.

5. Вопросы для самопроверки

1. Как строится проверочная матрица кодаХемминга?
2. Как строится порождающая матрица кодаХемминга?
3. Охарактеризуйте исправляющую способность кодов Хемминга.
4. Дайте математическое обоснование процедуры декодирования кодов Хемминга.
5. Каким образом вычисляются синдромы кодов Хемминга?
6. Каким образом обнаруживаются неисправимые ошибки кодом Хемминга?
7. Как задаются коды Рида-Соломона?
8. Охарактеризуйте исправляющую способность РС-кодов.
9. Дайте математическое обоснование процедуры декодирования РС-кодов.
10. Каким образом вычисляются синдромы РС-кода?
11. Как определяется коэффициенты полинома локаторов ошибок?
12. Как находятся корни степенных уравнений в конечных полях?
13. Каквычисляютсязначенияошибок?

6.Библиографический список

1. Блейхут Р. Теория и практика кодов, контролирующих ошибки [Текст] / Р.Блейхут. М.: Мир, 1986. 576 с.
2. Егоров С.И. Коррекция ошибок в информационных каналах периферийных устройств ЭВМ [Текст]: монография /С.И. Егоров: Курск.гос. техн. ун-т. Курск, 2008. 252 с.

Приложение 1. Варианты заданий на курсовой проект для проектирования кодека кода Хемминга.

	n	k	m	Расширенный код
1	15	11	4	нет
2	15	11	4	да
3	14	10	4	нет
4	14	10	4	да
5	13	9	4	нет
6	13	9	4	да
7	12	8	4	нет
8	12	8	4	да
9	11	7	4	нет
10	11	7	4	да
11	10	6	4	нет
12	10	6	4	да
13	9	5	4	нет
14	9	5	4	да
15	8	4	4	нет
16	8	4	4	да
17	7	3	4	нет
18	7	3	4	да
19	6	2	4	нет
20	6	2	4	да
21	16	11	5	нет
22	17	12	5	нет
23	18	13	5	нет
24	19	14	5	нет
25	20	15	5	нет

26	21	16	5	нет
27	22	17	5	нет
28	23	18	5	нет
29	24	19	5	нет
30	25	20	5	нет

Приложение 2. Варианты заданий на курсовой проект для проектирования кодека кода Рида-Соломона.

В строках таблицы приведены информационные символы слова кода Рида-Соломона (15,9). Необходимо закодировать слово и внести в него ошибки, указанные преподавателем. Код определен над полем $GF(2^4)$, $p(x)=x^4+x+1$. Порождающий многочлен кода $g(x)=(x- \alpha^x)(x- \alpha^{x+1})(x- \alpha^{x+2})(x- \alpha^{x+3})(x- \alpha^{x+4})(x- \alpha^{x+5})$, где x = номер варианта, взятый по модулю 15.

№	x^{14}	x^{13}	x^{12}	x^{11}	x^{10}	x^9	x^8	x^7	x^6	x^5	x^4	x^3	x^2	x^1	x^0
1	1	7	4	4	2	12	8	3	7						
2	5	14	12	1	-1	6	0	1	6						
3	6	1	2	12	8	3	7	1	12						
4	13	10	3	11	-1	5	6	1	9						
5	3	11	5	9	7	10	3	7	13						
6	9	11	0	4	4	9	11	2	1						
7	5	10	11	12	2	3	11	10	12						
8	3	10	14	8	4	1	3	10	11						
9	7	4	1	11	3	6	14	10	9						
10	13	11	6	2	10	1	2	13	9						
11	5	3	10	-1	6	14	1	13	4						
12	7	-1	4	11	13	2	4	3	5						
13	14	7	4	11	2	8	3	13	5						
14	9	5	0	12	5	6	8	0	1						
15	6	9	10	4	6	13	6	11	4						
16	7	3	13	8	-1	3	4	1	5						
17	3	8	5	0	12	5	14	8	4						
18	5	9	2	13	11	7	9	3	0						
19	9	11	7	0	-1	7	4	3	1						
20	1	8	11	14	2	9	0	3	-1						
21	5	7	4	4	-1	12	8	3	7						
22	6	14	12	1	8	6	0	1	6						
23	13	1	2	12	-1	3	7	1	12						

24	3	10	3	11	7	5	6	1	9						
25	9	11	5	9	4	10	3	7	13						
26	5	11	0	4	2	9	11	2	1						
27	3	10	11	12	4	3	11	10	12						
28	7	10	14	8	3	1	3	10	11						
29	13	4	1	11	10	6	14	10	9						
30	14	11	6	2	11	1	2	13	9						

Значения символов даны как степени α, -1 соответствует нулевому значению символа.

Необходимо исправить ошибки в указанном слове. Для нечетных номеров использовать определительный метод, для четных – алгоритм Берлекэмпа-Месси.