МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное
учреждение высшего профессионального образования

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
АЭРОКОСМИЧЕСКОГО ПРИБОРОСТРОЕНИЯ»**

КАФЕДРА №14

КУРСОВОЙ ПРОЕКТ
ЗАЩИЩЕН С ОЦЕНКОЙ

ПРЕПОДАВАТЕЛЬ

		М. О. Алексеев
должность, уч. ст., звание	подпись, дата	инициалы, фамилия

Пояснительная записка к курсовому проекту

«АНАЛИЗ ПОМЕХОУСТОЙЧИВОСТИ КАСКАДНОГО КОДА (БЧХ + СВЕРТОЧНЫЙ КОД)»

по курсу: «Кодирование и декодирование»

РАБОТУ ВЫПОЛНИЛ

СТУДЕНТ ГР.	5912		П. П. Морозкин
		подпись, дата	инициалы, фамилия

Санкт-Петербург
2013 г.

Содержание

1 Введение 4
2 Структура системы передачи информации 6
3 Теретические основы кодов, контролирующих ошибки 7
3.1 Блоковые коды 7
3.1.1 Линейные блоковые коды 8
3.1.2 Циклические коды 9
3.2 БЧХ-коды 11
3.2.1 Поиск порождающего многочлена 12
3.2.2 Выбор параметров кода 12
3.2.3 Построение конечного поля 13
3.2.4 Поиск примитивного элемента 13
3.2.5 Построение порождающего многочлена 14
3.2.6 Алгоритм кодирования БЧХ-кодом 15
3.2.7 Общий алгоритм декодирования 15
3.2.8 Алгоритм декодирования Берлекэмпа–Месси 15
3.2.9 Решение ключевого уравнения 17
3.2.10 Вычисление синдромов 20
3.2.11 Граница Синглтона 20
3.3 Сверточные коды 21
3.3.1 Алгоритм кодирования сверточным кодом 21
3.3.2 Алгоритм Витерби 22
3.4 Каскадные коды 23
3.4.1 Получение каскадных кодов 24
3.4.2 Декодирование каскадных кодов 24
4 Описание программной модели 26
4.1 Общие сведения 26
4.2 Используемый инструментарий и техника разработки 27
4.3 Архитектура модели 27
5 Запуск программной модели 29
6 Результаты моделирования 32
6.1 Передача изображения по системе моделирования 32
6.2 Оценка частоты появления ошибок на удаленной стороне 32
Заключение 41
Список использованных источников 42

Обозначения и сокращения

БЧХ-код — Код Боуза — Чоудхури — Хоквингема

1 Введение

В настоящий момент потребность в надежной передаче данных между устройствами остается актуальной. Это связано с тем, что в любом реальном физическом канале связи присутствуют посторонние шумы, искажающие поступающие данные. Для устранения искажений и получения оригинальной информации в настоящее время широко используется помехоустойчивое кодирование информации. Исходный поток данных разбивается на порции данных, которые кодируются помехоустойчивым кодом и передаются в закодированном виде в канал связи. После принятия искаженных данных из канала выполняется процесс декодирования полученной информации. В случае обнаружения ошибок используются алгоритмы, позволяющие восстановить оригинальную информацию. При использовании кодирования к исходным данным добавляется избыточная информация, которая позволяет на этапе декодирования определить наличие ошибок и выполнить их устранение.

Пионером теории кодирования приято считать Клода Шеннона, опубликовавшего в 1948 году статью, в которой он определил понятие *пропускной способности* канала. Было показано, что для каждого канала пропускная способность определяется числом и измеряется в битах в секунду. Затем было показано, что если скорость передачи данных R (бит/сек) меньше, чем пропускная способность канала, то с использованием помехоустойчивых кодов можно добиться сколь угодно малой вероятности ошибки на выходе. В дальнейшем многие исследователи приложили свои усилия для отыскания классов кодов, которые позволяют получить малую вероятность ошибки. В частности, одно из направлений было посвящено изучению *блоковых кодов*.

Первые блоковые коды были описаны Хеммингом в 1950 году. Он показал, что найденный им класс кодов позволяет строить коды, способные исправлять одиночные ошибки. Данное открытие стало прорывом в теории кодирования, и многие исследователи, воодушевленные работой Хемминга, продолжили поиски лучшего класса кодов. Однако, на протяжении десяти лет такой класс кодов получить не удалось. Но в 1960 году Боуз, Чоудхори и Хоквингем нашли класс кодов, позволяющих исправлять кратные ошибки. Полученные ими коды подучили название БЧХ-коды.

Другой класс кодов, позволяющий строить коды и кодировать ими информацию, получил название *сверточных кодов*. Развитие сверточных кодов заметно отличалось от развития теории блоковых кодов. Разницу можно почуствовать даже если сравнить подходы для нахождения хороших классов кодов. Так, при построении блоковых кодов и создании эффективных алгоритмов декодирования широко используются *алгебраические методы*. Со сверточными кодами ситуация иная. Например, хорошие сверточные коды были найдены с использованием вычислительной техники. Для этого выполнялся анализ большого числа кодов, а затем выбирались коды с лучшими характеристиками.

Другим отличительным свойством сверточных кодов является отсутствие методов декодирования, аналогичных алгебраическим методам исправления кратных ошибок. Наиболее используемым методом декодирования сверточных кодов является *метод максимального правдоподобия*, на основе которого функционирует *алгоритм Витерби*. Сверточное кодирование и декодирование с использованием алгоритма Витерби стало широко применяться на практике. Например, указанное кодирование применяется при передаче денных в космическом пространстве и при передаче денных из космического пространства на Землю и обратно. Причина, по которой данное кодирование столь распространено, состоит в относительной простоте реализации и в достижении выигрыша при декодировании.

Для улучшения характеристик системы передачи информации было предложено использовать *каскадные коды*. В этом случае исходных поток данных после его разбиения на порции данных кодируется одним их блоковых кодов. Широко используемым кодом для данной цели является БЧХ-код. Далее полученное кодовое слово БЧХ-кода кодируется сверточным кодом. Полученное слово сверточного кода отправляется в канал. Таким образом два кодера работают в каскаде, откуда и произошло характерное название данного класса кодов. При декодировании каскадного кода выполняется ряд обратных кодированию процедур. Так, на первом этапе с использованием алгоритма Витерби происходит декодирование данных, поступивших из канала. В результате декодирования имеем кодовое слово БЧХ-кода, которое дополнительно декодируется БЧХ- декодером. Особая эффективность такого способа кодирования состоит в том, что характерные свойства как БЧХ-кодов, так и сверточных кодов (и алгоритма Витерби) используются совместно, что позволяет достигать более высоких показателей, таких как вероятность ошибки на бит. Недостатком такого подхода является, пожалуй, относительная низкая скорость кода.

2 Структура системы передачи информации

Система передачи информации показана на Рис. 2.1. Она состоит из источника информации, генерирующего исходный поток данных. Источник информации передает данные в кодер источника информации, который предназначен для устранения естественной избыточности генерируемой информации. Кодированные данные затем поступают в кодер канала. Данный компонент предназначен для кодирования данных помехоустойчивым кодом для возможности обнаружения ошибок на удаленной стороне и восстановления исходных данных.

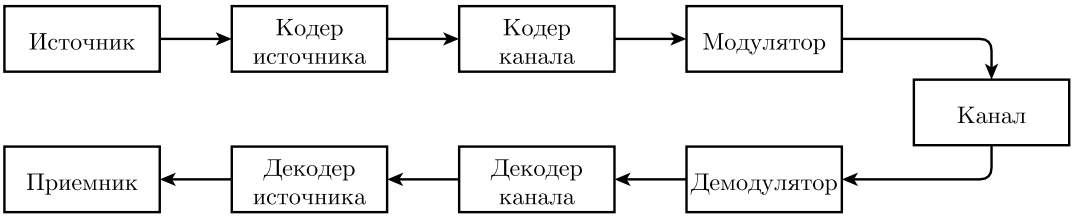

Рис. 2.1 — Структура системы передачи информации

Таким образом, кодированные данные поступают в модулятор, который предназначен для генерации по цифровым данным реальных физических сигналов и их передачу в канал. Причем в данной ситуации каналом может являться как телефонный кабель, оптическое волокно, воздушная среда, так и система хранения информации, например, жесткий диск, магнитный диск или флеш- накопитель. По сути, все эти устройства реализуют канал связи, который в общих чертах можно охарактеризовать как компонент, позволяющий передавать данные и обладающий тем свойством, что при передаче данных могут возникать ошибки. Их появление объясняется природой канала и не может быть определено заранее. После передачи данных каналом и получения данных на удаленной стороне начинается обратный процесс, нацеленный на восстановление переданных данных. Вначале данные из канала демодулируются демодулятором, который генерирует кодовые слова выбранного кода. Данные слова декодируются декодером канала. В случае обнаружения декодером ошибок при передаче данных, выполняется процесс устранения данных ошибок с целью получить исходные денные. В результате декодирования данные поступают в декодер источника. На этом этапе выполняется декодирование данных, изначально сгенерированных источником, добавляется избыточная информация. Полученные денные отправляются приемнику информации.

3 Теретические основы кодов, контролирующих ошибки

В данном разделе приводятся сведения о кодах, контролирующих ошибки, необходимые для дальнейшего изложения материала.

3.1 Блоковые коды

Определение 3.1.1 Блоковый код *мощности M над алфавитом из q символов — это множество из M q-ичных последовательностей длины n, называемых* кодовыми словами.

При $q = 2$ символы называются ***битами***, а код — ***двоичным***. Как правило, $M = q^k$ для некоторого k. Такой код носит название (n,k)-*код*. Каждой последовательности из k q-ичных символов можно сопоставить последовательность из n q-ичных символов, которая и является кодовым словом. Блоковый код задает n-символьное кодовое слово, которое включает k информационных символов.

Определение 3.1.2 Скорость *блокового кода — величина $R = k/n$.*

Основные параметры блокового кода:

а) длина блока n

б) информационная длина k

в) минимальное расстоянию d^*

Минимальное расстояние является мерой различия двух наиболее похожих кодовых слов.

Определение 3.1.3 Расстояние по Хеммингу *между двумя q-ичными последовательностями x и y длины n — число позиций, в которых данные последовательности отличаются. Обозначение: $d(x, y)$.*

Пример расстояния, $d(01011, 00111) = 2$.

Определение 3.1.4 *Пусть $\mathfrak{C} = \{c_i, i = 0, \ldots, M-1\}$ — код. Тогда* **минимальное расстояние** *кода $\mathfrak{C}$ равно наименьшему из всех расстояний по Хеммингу между различными парами кодовых слов, т.е.*

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

Определение 3.1.5 *(n, k)-код с минимальным расстоянием d^* называется (n, k, d^*)-кодом.*

В случае возникновения в канале t ошибок при передаче кодового слова результат декодирования зависит от расстояния от принятого слова до каждого другого слова. Так, в случае, если расстояние от принятого слова до каждого другого кодового слова превосходит t, то ошибки будут исправлены, а ближайшее к принятому кодовое слово будет принято в качестве действительно переданного. Данное утверждение выполняется только в случае

$$d^* \geqslant 2t + 1.$$

3.1.1 Линейные блоковые коды

Большинство известных хороших кодов принадлежат классу ***линейных кодов***.

Определение 3.1.6 Линейный код — *подпространство в векторном пространстве* $GF^n(q)$.

Таким образом, линейный код есть непустое множество n-последовательностей над полем Галуа $GF(q)$. Данные последовательности называемых кодовыми словами. Особенностью множества является то, что сумма двух кодовых слов является кодовым словом, а произведение любого кодового слова на элемент поля также является кодовым словом.

Определение 3.1.7 Систематический код — *код, у которого всякое кодовое слово начинается с информационных символов. Оставшиеся символы называются* **проверочными символами**.

Определение 3.1.8 Вес Хемминга $\omega()$ *кодового слова* c — *число ненулевых компонент.* **Минимальный вес кода** ω^* — *минимальный вес ненулевого кодового слова.*

Теорема 3.1.1 *Для линейного кода минимальное расстояние* d^* *равно минимальному весу* ω^*.

Теорема 3.1.2 *Минимальное расстояние любого линейного* (n, k)*-кода удовлетворяет неравенству*

$$d^* \leqslant 1 + n - k.$$

Определение 3.1.9 *Любой* (n, k)*-код с минимальным расстоянием, которое удовлетворяет равенству*

$$d^* = 1 + n - k$$

называется **кодом с максимальным расстоянием**.

Согласно Границе Синглтона для исправления t ошибок код должен иметь не менее $2t$ проверочных символов (2 проверочных символа на ошибку). Коды с максимальным расстоянием имеют точно $2t$ проверочных символов.

3.1.2 Циклические коды

Циклические коды составляют относительно большую группу наиболее широко используемых на практике линейных систематических кодов. Их основное свойство, давшее им название, заключается в том, что каждый вектор, получаемый из исходного кодового вектора путем циклической перестановки его символов, также является разрешенным кодовым вектором. В качестве математического аппарата для построения данных кодов используется теория полей Галуа.

Операции кодирования и декодирования циклических кодов сводятся к известным процедурам умножения и деления полиномов. Для двоичных кодов эти операции легко реализуются технически с помощью линейных переключательных схем (ЛПС), при этом получаются относительно простые схемы кодеков, в чем состоит одно из практических достоинств циклических кодов.

Определение 3.1.10 *Линейный* (n, k)*-код* $\mathfrak{C}$ *называется* **циклическим**, *если из того, что слово* $c = (c_0, c_1, \ldots, c_{n-1})$ *принадлежит коду* $\mathfrak{C}$, *следует, что слово* $c' = (c_{n-1}, c_0, c_1, \ldots, c_{n-2})$ *также принадлежит коду* $\mathfrak{C}$.

Кодовое слово c' получается циклическим сдвигом всех компонент слова c вправо на одну позицию. Каждый линейный код над $GF(q)$ длины n представляет собой подпространство пространства $GF^n(q)$, а циклический код является частным случаем подпространства, так как обладает дополнительным свойством цикличности.

Каждый вектор из $GF^n(q)$ можно представить многочленом от x степени не выше $n-1$. Компоненты вектора отождествляются с коэффициентами многочлена. Множество многочленов обладает структурой векторного пространства, идентичной структуре пространства $GF^n(q)$. Это же множество многочленов обладает структурой кольца $GF(q)[x]/(x^n-1)$. Как в кольце, в этом множестве определено умножение

$$p_1(x) \cdot p_2(x) = R_{x^n-1}\left[p_1(x)p_2(x)\right]$$

где $R_{h(x)}\left[f(x)g(x)\right]$ — есть остаток от деления на многочлен $h(x)$ произведения многочленов $f(x)$ и $g(x)$. Заметим, что в приведенное равенство входят произведения двух видов. Произведение в левой части является произведением в кольце $GF(q)[x]/(x^n-1)$, определенным через произведения к кольце $GF(q)[x]$ в правой части. Циклический сдвиг может быть записан через умножение в этом кольце:

$$x \cdot p(x) = R_{x^n-1}[xp(x)].$$

Итак, если кодовые слова некоторого кода задаются в виде многочленов, то код является подмножеством кольца $GF(q)[x]/(x^n-1)$. Такой код является циклическим, если вместе с каждым кодовым словом $c(x)$ он содержит кодовый многочлен $x \cdot c(x)$.

Определение 3.1.11 Порождающий многочленом *кода* $\mathfrak{C}$ *— единственный приведенный ненулевой многочлен наименьшей степени в коде* $\mathfrak{C}$*. Обозначение:* $g(x)$*.*

Теорема 3.1.3 *Циклический* (n, k)*-код состоит из всех произведений порождающего многочлена* $g(x)$ *степени* $n - k$ *на многочлены степени не выше* $k - 1$*.*

Теорема 3.1.4 *Циклический код длины* n *с порождающим многочленом* $g(x)$ *существует тогда и только тогда, когда* $g(x)$ *делит* $x^n - 1$*.*

Согласно теореме 3.1.4, для порождающего многочлена $g(x)$ любого циклического кода выполняется равенство

$$x^n - 1 = g(x)h(x)$$

при некотором многочлене $h(x)$. Многочлен $h(x)$ называется ***проверочным многочленом***. Каждое кодовое слово $c(x)$ удовлетворяет равенству

$$R_{x^n-1}[h(x)c(x)] = 0.$$

Пусть $c(x)$ обозначает переданное кодовое слово. Это значит что символами переданного слова были коэффициенты многочлена $c(x)$. Пусть многочлен $v(x)$ обозначает принятое слово, и пусть $e(x) = v(x) - c(x)$. Многочлен $e(x)$ называется ***многочленом ошибок***. Ненулевые коэффициенты этого многочлена стоят в тех позициях, где в канале произошли ошибки.

Представим информационную последовательность в виде многочлена $i(x)$ степени не выше $k-1$. Множество информационных многочленов можно отобразить в кодовые многочлены многими способами. Одним простым правилом является

$$c(x) = i(x)g(x).$$

Такой кодер является несистематическим, так как по многочлену $c(x)$ нельзя сразу установить $i(x)$. Систематическое правило кодирования имеет следующий вид:

$$c(x) = x^{n-k}i(x) + t(x),$$

где $t(x) = -R_{g(x)}\left[x^{n-k}i(x)\right]$.

И систематическое, и несистематическое правила кодирования дают одно и тоже множество кодовых слов, но соответствия между $i(x)$ и $c(x)$ различны.

Определение 3.1.12 Синдромный многочлен $s(x)$ *— остаток от деления многочлена* $v(x)$ *на* $g(x)$*:*

$$s(x) = R_{g(x)}[v(x)] = R_{g(x)}[c(x) + e(x)] = R_{g(x)}[e(x)].$$

3.2 БЧХ-коды

Коды Боуза-Чоудхури-Хоквингема (БЧХ) составляют один из больших классов линейных кодов, исправляющих ошибки. Причем метод построения этих кодов задан явно. Интерес к кодам БЧХ определяется тем, что они позволяют исправлять любое наперед заданное число ошибок и для них существуют эффективные алгоритмы кодирования и декодирования.

Порождающий многочлен циклического кода можно представить в виде

$$g(x) = [f_1(x), f_2(x), \ldots, f_r(x)],$$

где $f_i(x), i = 1, \ldots, r$ — минимальные многочлены корней $g(x)$.

Пусть элементы $GF(q^m)$ $\gamma_1, \gamma_2, \ldots, \gamma_r$ — корни порождающего многочлена $g(x)$. Тогда

$$v(\gamma_i) = c(\gamma_i) + e(\gamma_i) = e(\gamma_i) = \sum_{j=0}^{n-1} e_j \gamma_i^j.$$

В результате получаем r уравнений, содержащих только величины, определяемые ошибками и не зависящие от кодового слова. Если эти уравнения можно разрешить относительно e_j, то мы сможем определить многочлен ошибок. Нужно выбрать γ_i таким образом, чтобы система r уравнений могла быть решена относительно e_i каждый раз, когда не более t неизвестных отличны от нуля.

Для произвольного циклического кода с порождающим многочленом $g(x)$, имеющим корни $\gamma_1, \ldots, \gamma_r$, определим компоненты синдрома

$$S_j = v(\gamma_j), j = 1, \ldots, r.$$

Эти элементы поля отличны от синдромного многочлена $s(x)$, но содержат эквивалентную информацию. Мы хотим подобрать $\gamma_1, \ldots, \gamma_r$ так, чтобы по $S_1, \ldots, S_r$ можно было найти t ошибок. В качестве таких γ_i можно взять степени $\{\alpha, \alpha^2, \ldots, \alpha^{2t}\}$ примитивного элемента α поля $GF(q^m)$.

Определение 3.2.1 *Пусть заданы q и m, и пусть β — любой элемент поля $GF(q^m)$ порядка n. Тогда для любого положительного целого числа t циклическим кодом длины n с порождающим многочленом*

$$g(x) = (f_1(x), \ldots, f_{2t}(x)),$$

где $f_j(x)$ — минимальный многочлен элемента β^j.

Определение 3.2.2 Примитивный *БЧХ-код — код длины $q^m - 1$.*

Для того, чтобы построить порождающий многочлен примитивного БЧХ-кода нужно:

а) Задать длину кода $n = q^m - 1$ и число t ошибок, которые необходимо исправлять.

б) Найти неприводимый многочлен степени m и построить поле $GF(q^m)$.

в) Найти примитивный элемент α в поле $GF(q^m)$.

г) Найти минимальные многочлены $f_i(x)$ для $\alpha^i, i = 1, \ldots, 2t$ над $GF(q)$.

д) Взять в качестве $g(x) = (f_1(x), f_2(x), \ldots, f_{2t}(x))$.

3.2.1 Поиск порождающего многочлена

Для того, чтобы найти порождающий многочлен БЧХ, надо выполнить следующие шаги:

а) Выбрать основание — простое число p.

б) Выбрать число членов поля $q = pk$, над которыми будет построен код БЧХ, где k — натуральное число.

в) Определить длину кода n, дина кода n определяется из формулы $n = \frac{q^m-1}{s}$, где m, s — натуральные числа.

г) Задать желаемое минимальное расстояние для кода d, значение минимального расстояния кода d должно быть меньше длины кода n.

д) Построить поле $GF(q)$.

е) Построить поле $GF(q^m)$, где m — параметр кода БЧХ.

ж) Найти примитивный элемент λ поля $GF(q^m)$.

з) Найти степень примитивного элемента $\beta = \lambda\ s$, где s — параметр кода БЧХ.

и) Найти $d-1$ последовательные степени $\beta : \beta^l, \beta^{l+1}, \beta^{l+2}, \ldots \beta^{l+d-2}$, где l — произвольное натуральное число.

к) Найти нормированный многочлен $g(x)$ минимальной степени над полем $GF(q)$, корнями которого являются все $d\breve{}1$ подряд идущих степеней $\beta^l, \beta^{l+1}, \beta^{l+2}, \ldots \beta^{l+d-2}$.

л) Этот многочлен $g(x)$ и является порождающим для БЧХ-кода с заданными выше параметрами.

3.2.2 Выбор параметров кода

Параметры кода БЧХ выбираются исходя из требований задачи, в которой код будет применен. Ниже будет использоваться $p = 2$ и $k = 1$. Последнее обусловлено тем, что в основном коды БЧХ применяют в вычислительных устройствах, которые используют двоичное представление чисел и, следовательно, удобнее использовать двоичные коды БЧХ.

3.2.3 Построение конечного поля

Существует два варианта построения поля, в зависимости от количества элементов.

а) Поле содержит p элементов, где p — простое. В данном случае полем является кольцо вычетов по модулю p.

б) Поле содержит $q = pk$ элементов, где p — простое, k — натуральное числа. Для построения поля из $q = pk$ элементов достаточно отыскать многочлен $f(x)$ степени k, неприводимый над полем $GF(p)$.

В случае кодов БЧХ оба поля, используемые в построении, являются полями $GF(q)$ и $GF(q^m)$, т.е. для их построения необходимо отыскание двух неприводимых многочленов — степени k и степени km над полем $GF(p)$.

3.2.4 Поиск примитивного элемента

В построенном поле $GF(q^m)$ находим примитивный элемент α. Находим его методом перебора, т.к. других методов поиска примитивного элемента пока не найдено. Каждый элемент поля возводим в степени от 1 до q и проверяем полученные элементы на равенство каждому из элементов поля. В том случае, если из проверяемого элемента путем возведения его в натуральные степени получаем все элементы поля, то проверяемый элемент — примитивный. Возводим α в степень s, находим $\beta = \alpha^s$. Значение β возводим в $d-1$ последовательные степени $\beta : \beta^l, \beta^{l+1}, \beta^{l+2}, ...\beta^{l+d-2}$, где l — произвольное натуральное число. Теперь надо найти такой многочлен $g(x)$ минимальной степени над полем $GF(q)$, что все $\beta^l, \beta^{l+1}, \beta^{l+2}, ...\beta^{l+d-2}$ будут его корнями. Проще всего сделать это следующим образом:

а) Построить циклотомические классы для всех $\beta^l, \beta^{l+1}, \beta^{l+2}, ...\beta^{l+d-2}$.

б) Найти многочлен для каждого циклотомического класса.

в) Порождающий многочлен $g(x)$ можно найти как произведение многочленов циклотомических классов.

Циклотомическим классом элемента β поля $GF(q^m)$ над полем $GF(q)$ называют все различные элементы поля $GF(q^m)$, порожденные последовательным возведением β в степень q : $\beta, \beta^q, \beta^{q^2}, \beta^{q^3}, ...$ Заметим, что циклотомических классов может быть меньше, чем $d-1$, т.к. некоторые степени β могут входить в один и тот же циклотомический класс. Многочлен циклотомического класса — многочлен минимальной степени, корнями которого являются все элементы циклотомического класса. Многочлен циклотомического класса имеет коэффициентами элементы поля $GF(q)$, несмотря на то, что строится над полем $GF(q^m)$.

3.2.5 Построение порождающего многочлена

После построения многочленов циклотомических классов легко вычислить порождающий многочлен. Порождающий многочлен БЧХ есть произведение многочленов всех циклотомических классов. Перемножив эти многочлены получаем многочлен $g(x)$, порождающий БЧХ-код над полем $GF(q)$. Часто БЧХ-код записывают в виде трех чисел: (n, l, d), где n — длина кодового слова БЧХ, l — длина исходного кодового слова, d — минимальное кодовое расстояние.

Более строгое определение гласит. БЧХ-код с кодовым расстоянием $d_{min} \leq 2_{t_d} + 1$ является циклическим кодом, порождающим многочлен $g(x)$, который имеет $2_{t_d} + 1$ последовательных корней в точках $\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta}$, где $\delta = 2_{t_d} - 1$.

Таким образом, порождающий многочлен двоичного (n, k, d_{min}) кода БЧХ имеет вид

$$g(x) = \{\phi_b(x), \phi_{b+1}(x), \ldots, \phi_{b+2t_d-1}(x)\}.$$

Пример В поле $GF(2^4)$, порождаемом примитивным многочленом $p(x) = x^4 + x + 1$, и параметры $t_d = 3, b = 1$, многочлен

$$g(x) = \{\phi_1(x), \phi_3(x), \phi_5(x)\} = (x^4+x+1)(x^4+x^3+x^2+x+1)(x^2+x+1) = x^{10}+x^8+x^5+x^4+x^2+x+1$$

порождает двоичный $(15, 5, 7)$ код БЧХ, исправляющий три ошибки.

Рассмотрим нижнюю границу минимального расстояния кодов БЧХ, известную как *граница БЧХ*. Элементы $\alpha^b, \alpha^{b+1}, \ldots, \alpha^{b+\delta}$, де $\delta = 2_{t_d} - 1$, являются корнями порождающего многочлена $g(x)$ и что все кодовые слова кода БЧХ, ассоциированные с полиномами $v(x)$ кратны порождающему многочлену кода. Следовательно

$$v(x) \in C \Leftrightarrow v(\alpha') = 0, b \leq i \leq b + 2_{t_d} - 1$$

Таким образом, все кодовые слова удовлетворяют следующей системе 2_{t_d} уравнений (в матричной форме)

$$(v_0, v_1, \ldots, v_{n-1}) \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha^b & \alpha^{b+1} & \cdots & \alpha^{b+2t_d-1} \\ \alpha^{2b} & \alpha^{2(b+1)} & \cdots & \alpha^{2(b+2t_d-1)} \\ \vdots & \vdots & \ddots & \vdots \\ \alpha^{(n-1)b} & \alpha^{(n-1)(b+1)} & \cdots & \alpha^{(n-1)(b+2t_d-1)} \end{pmatrix} = \mathbf{0}$$

Соответственно, проверочная матрица двоичного БЧХ кода имеет вид

$$\mathbf{H} = \begin{pmatrix} 1 & \alpha^b & \alpha^{2b} & \cdots & \alpha^{(n-1)b} \\ 1 & \alpha^{b+1} & \alpha^{2(b+1)} & \cdots & \alpha^{(n-1)(b+1)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha^{b+2t_d-1} & \alpha^{2(b+2t_d-1)} & \cdots & \alpha^{(n-1)(b+2t_d-1)} \end{pmatrix}$$

Эта проверочная матрица обладает свойством: любая ее $2_{t_d} * 2_{t_d}$ подматрица является *матрицей Вандермонда*. Следовательно, любые 2_{t_d} столбцов матрицы линейно независимы. Отсюда следует, что минимальное кодовое расстояние этого кода удовлетворяет неравенству $d \geq 2_{t_d} + 1$.

3.2.6 Алгоритм кодирования БЧХ-кодом

Кодирование БЧХ-кодом состоит из следующих операций:

а) Получить вектор бит для кодирования.

б) Преобразовать данный вектор в полином $m(x)$.

в) Умножить полученный полином $m(x)$ на порождающий полином $g(x)$. В результате имеем полином $r(x)$.

г) Преобразовать полученный полином $r(x)$ в вектор бит.

д) Данный вектор бит явялется кодовым словом БЧХ-кода и передается в канал.

3.2.7 Общий алгоритм декодирования

На Рис. 3.1 показана обобщенная структура декодера циклического кода. Синдром $s(x)$ используется для определения полинома ошибок $e(x)$. Проблема декодирования равноценна поиску полинома ошибок $e(x)$ по известному синдрому $s(x)$.

Рис. 3.1 — Обобщенная структура декодера циклического кода

3.2.8 Алгоритм декодирования Берлекэмпа–Месси

Декодирование БЧХ-кодов распадается на несколько задач. Это (1) вычисление синдромных компонент, (2) отыскание полинома локаторов ошибок, (3) отыскание самих локаторов ошибок как величин, обратных корням полинома локаторов ошибок, (4) определение величин ошибок. Архитектура БЧХ-декодера показана на Рис. 6.22.

Первоначальная версия алгоритма Берлекэмпа–Месси была изложена Берлекэмпом в 1968 году в качестве элемента конструкции декодера кодов Боуза–Чоудхудри–Хоквингема над конечным полем. Хотя в этой работе была указана

Рис. 3.2 — Архитектура БЧХ-декодера

возможность формулировки решаемой задачи с использованием понятия линейного регистра сдвига с обратной связью, алгоритм описывался исключительно в терминах полиномов и был весьма сложен для понимания. Спустя год Месси предложил свою интерпретацию алгоритма, как позволяющего строить линейный регистр сдвига минимальной длины, генерирующий заданную последовательность. Эта интерпретация оказалась полезной для более широкого распространения алгоритма, получившего название по имени этих двух ученых. В некоторых работах алгоритм излагается также с помощью непрерывных дробей и рациональной аппроксимации.

Данный алгоритм по числу операций в поле обладает высокой эффективностью и обычно рассматривается как итеративный процесс построения минимального линейного регистра сдвига с обратной связью. Особенностью данного регистра является то, что он генерирует заранее известную *последовательность синдромов* $S_1, S_2, ..., S_{2d}$.

Алгоритм Берлекэмпа–Месси нацелен на построение многочлена $\sigma^{(l+1)}(x)$ наименьшей степени, который удовлетворяет уравлению:

$$\sum_{j=0}^{l_i+1} S_{k-j}{\sigma_j}^{(i+1)} = 0, l_i < k < i+1$$

Решение денной задачи эквивалентно условию, что многочен

$${\sigma}^{(i+1)}(x) = 1 + {\sigma_1}^{(i+1)}x + \ldots + {\sigma_{l_i+1}}^{(i+1)}x^{l_{i+1}}$$

является многочленом обратной связи ЛРОС, который генерирует ограниченную последовательность синдромов. Различие на i-й итерации определяется как

$$d_i = S_{i+1} + S_i\sigma_1^{(i)} + \ldots + S_{i-l_i+1}\sigma_{l_i}^{(i)}$$

является мерой соответствия синдромной последовательности и генерируемой ЛРОС и содержит корректирущий множитель для вычисления $\sigma^{(i+1)}$ на следующей итерации. Существуют два случая:

а) Если $d_i = 0$, то уравнение удовлетворяется неравенством.

$$\sigma(i+1)(x) = \sigma^1(x), l_{i+1} = l_i$$

б) Если $d \neq 0$, то решение на следующей итерации имеет вид

$$\sigma^{(i+1)}(x) = \sigma^1(x) + d_i {d_m}^{-1} x^{t-m} \sigma^{(m)}(x)$$

$$l_{i+1} = max\{l_i, l_m + i - m\},$$

где является решением на m-й итерации такое, что $-1 \leq m < i$, $d_m \neq 0$, и разность $(m - l_m)$ максимальна. Итеративное вычисление $\sigma^{(i+1)}(x)$ продолжается, пока не удовлетворится одно или оба условия: либо $i \geq l_{i+1} + t_d - 1$, либо $i = 2t_d - 1$.

Начальные условия алгоритма:

$$\sigma^{(-1)}(x) = 1, l_{-1} = 0, d_{-1} = 1$$

$$\sigma^{(0)}(x) = 1, l_0 = 0, d_0 = S_1.$$

Алгоритм Берлекэмпа–Месси можно рассматривать как итеративный процесс построения минимального линейного регистра сдвига с обратной связью (ЛРОС), аналогично показанному на Рис. 6.23, который генерирует известную *последовательность синдромов* $S_1, S_2, \ldots, S_{2t_d}$.

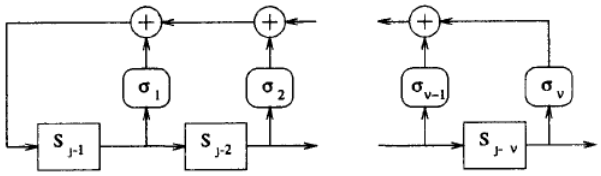

Рис. 3.3 — Архитектура БЧХ-декодера

3.2.9 Решение ключевого уравнения

Если кодер передает слово двоичного БЧХ-кода

$$C(x) = \sum_{i=0}^{n-1} C_i x^i,$$

а шум в канале задается вектором

$$E(x) = \sum_{i=0}^{n-1} E_i x^i,$$

то полученное слово записывается многочленом

$$R(x) = \sum_{i=0}^{n-1} R_i x^i = \sum_{i=0}^{n-1} C_i x^i + \sum_{i=0}^{n-1} E_i x^i.$$

Для $j = 1, 2, \ldots, 2t$ кодовое слово кратно минимальному многочлену элемента α^j и, следовательно,

$$R(\alpha^j) = 0 + \sum_{i=0}^{n-1} E_i \alpha^{ji} = S_j,$$

где элементы поля Галуа $X_1, X_2, \ldots, X_e$ – локаторы ошибок $E_i = 1$. Так как $S_j = R(\alpha^j) = r^{(\alpha^j)}(\alpha^j)$, где $r^j(x)$ – остаток от деления $R(x)$ на минимальный многочлен $M^{(j)}(x)$ элемента $\alpha^j (1 \leq j \leq 2t)$. После вычисления $S_1, S_2, \ldots, S_{2t}$ основная задача декодера – определить $X_1, X_2, \ldots, X_e$ из уравнений

$$\sum_{i=1}^{e} X_i^j = S_j, j = 1, 2, \ldots, 2t.$$

В обшщем случае эта система имеет много решений, каждое из которых соответствует различным векторам ошибок, лежащим в одном классе смежности по аддитивной группе кодовых слов. Декодер должен найти решение с наимешьшим возможным числом e.

Для решения системы декодер сначала пытается определить коэффициенты многочлена локаторов ошибок.

Определение 3.2.3

$$\sigma(z) = \prod_{i=1}^{e}(1 - X_i z) = 1 + \sum_{j=1}^{e} \sigma_j z^j.$$

Если многочлен $\sigma(z)$ уже найден декодером, то, используя процедуру Ченя, можно найти взаимные корни для $\sigma(z)$. Наиболее тяжелая часть этой процедуры – определение коэффициентов σ_j по величинам S_j.

Для того чтобы найти зависимость между величинами σ_j и S_j, введем производящую функцию

$$S(z) = \sum_{i=1}^{\infty} S_j z^j = \sum_{j=1}^{\infty} \sum_{i=1}^{e} X_i^j z^j = \sum_{i=1}^{e} \frac{X_{i^z}}{1 - X_{i^z}}.$$

После умножения на $\sigma(z)$ имеем

$$S(z)\sigma(z) = \sum_{i=1}^{e} \frac{X_{i^z}}{1 - X_{i^z}} \prod_{j=1}^{e}(1 - X_{i^z}) = \sum_{i=1}^{e} X_{(}i^z) \prod_{j \neq i}(1 - X_{i^z}).$$

Прибавив к обеим частям равенства $\sigma(z)$ получим

$$[1 + S(z)]\sigma(z) = \sigma(z) + \sum_{i=1}^{e} X_{i^z} \prod_{j \neq i}(1 - X_{t^z}).$$

Определим многочлен $\omega(z) = \sum\limits_{k=0}^{e} \omega_k z^k$ спомощью равенства

$$\omega(z) = \sigma(z) + \sum_{k=0}^{e} X_{i^z} \prod_{j \neq i}(1 - X_{t^z}).$$

Тогда

$$[1 + S(z)]\sigma(z) = \omega(z).$$

В общем случае декодеру известны коэффициенты только при первых $2t$ степенях z в $S(z)$ и не известны коэффициенты $S_{2t+1}, S_{2t+2}, S_{2t+3}, \ldots$. Иными словами, декодер не знает $S(z)$, но знает $S(z) mod z^{2t+1}$. Поэтому естественно ввести уравнение, которое мы назовем *ключевым*:

$$[1 + S(z)]\sigma(z) = \omega(z) mod z^{2t+1}.$$

Из этого уравнения по заданному $S(z)$ нужно найти *оба* многочлена $\sigma(z)$ и $\omega(z)$, степени которых не превосходят e, где e – число ошибок в канале.

Одна «физическая интерпретация» ключевого уравнения была предложена Месси. Перепишем уравнение в виде

$$S_k + \sum_{i=1}^{k-1} \sigma_i S_{k-i} + \sigma_k = \omega_k,$$

или в виде

$$S_k = \omega_k - \sum_{i=1}^{k-1} \sigma_i S_{k-i} - \sigma_k.$$

Последнее равенство задает k-й выход регистров сдвига, приведенных на Рис. 6.17 и Рис. 6.18. Обратные связи которых соответствуют многочлену $\sigma(z)$, а начальное состояние ячеек – многочлену $\omega(z)$.

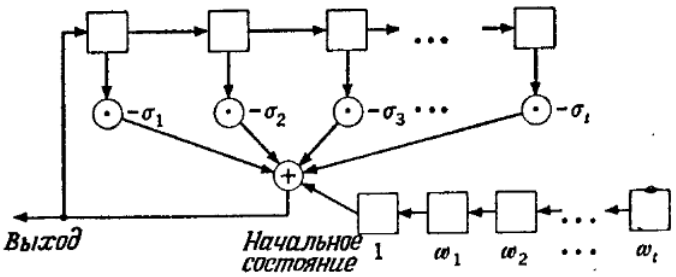

Рис. 3.4 — Интерпретация ключевого уравнения с помощью РОС

Это позволяет интерпретировать ключевое уравнение как математическую постановку задачи синтеза регистра с обратной связью: по заданной выходной последовательности $1 + S(z)$ необходимо определить обратные связи $\sigma(z)$ и начальное состояние $\omega(z)$ кратчайшего регистра сдвига с выходной последовательностью $1+S(z)$.

В задаче декодирования двоичных БЧХ-кодов реальный интерес представляет многочлен $\sigma(z)$, а не многочлен $\omega(z)$. Однако это же ключевое уравнение возникает и в некоторых других приложениях.

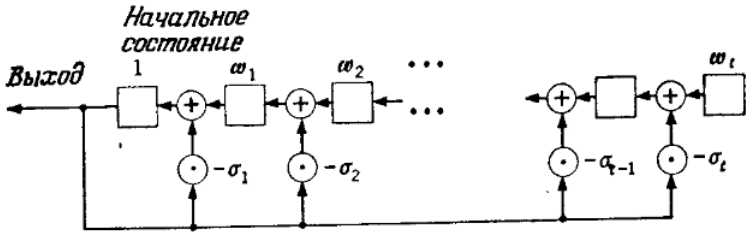

Рис. 3.5 — Другая интерпретация ключевого уравнения с помощью РОС

3.2.10 Вычисление синдромов

Синдромы вычисляются как значения принятого полнинома в нулях кода [1]

$$S_i = r(\alpha^i), i = b, b+1, \ldots, b+2t_d-1$$

Вычисление синдромов может быть реализовано аппаратно. На Рис. 6.21 показан пример схемы аппаратного вычисления синдрома.

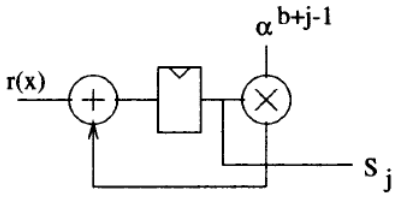

Рис. 3.6 — Пример вычисления синдрома

3.2.11 Граница Синглтона

Определение

Граница Синглтона (названная в честь Р. К. Синглтона) устанавливает предел мощности кода C с символами из поля $\mathbb{F}_q$ длины n и минимального расстояния Хэмминга d.

Пусть $A_q(n,d)$ обозначает максимально возможную мощность q-ичного кода длины n (q-ичный код — это код над полем из q элементов). Пусть минимальное расстояние Хэмминга между двумя словами кода будет d, то есть $\mathrm{D}_H(w, w') \geqslant d$ для любых двух кодовых слов w и w'.

Тогда : $A_q(n, d) \leqslant q^{n-d+1}$.

Доказательство

В первую очередь заметим, что верхняя граница максимальной мощности любого q-ичного кода длины n равняется q^n, так как каждый компонент данного

[1] Для двоичных БЧХ кодов справедливо равенство $S_{2i} = S_i^2$, которое позволяет сократить объем вычислений.

кодового слова может принимать одно из q разных значений независимо от других компонентов.

Пусть C является q-ичным кодом. Тогда все слова $c \in C$ в коде отличны друг от друга. Если мы сотрём первые $d-1$ символов каждого слова, тогда все оставшиеся кодовые слова должны оставаться разными, так как расстояние Хэмминга между словами кода C по меньшей мере d. Следовательно мощность кода после удаления $d-1$ символов осталась прежней.

Длина нового кода

$$n - (d-1) = n - d + 1,$$

и следовательно максимально возможной мощностью такого кода является q^{n-d+1}.

Отсюда следует верхняя граница мощности и для изначального кода

$$A_q(n, d) \leqslant q^{n-d+1}.$$

Линейные коды

В случае с линейными кодами можно записать границу Синглтона как

$$q^k \leqslant q^{n-d+1}$$

или

$$k \leqslant n - d + 1.$$

Линейные коды, для которых выполняется равенство $k = n - d + 1$, называются *разделимыми кодами с максимальным расстоянием* или кодами МДР. Известными представителями этого семейства кодов являются *код Рида — Соломона* и коды, образуемые из него.

3.3 Сверточные коды

Определение 3.3.1 Сверточные кодирование — *метод кодирования, при котором каждый символ входной последовательности, состоящей из k битов, преобразуется в n-битовый кодированный поток данных. Внесение избыточности и соответственное увеличение скорости передачи в n/k раз, позволяет повысить помехоустойчивость, особенно в каналах с одиночными ошибками.*

3.3.1 Алгоритм кодирования сверточным кодом

Алгоритм основан на работе сверточного кодера. Кодер имеет набор входов и выходов. Символы информационной последовательности поступают на вход кодера и попадают в регистр сдвига с обратной связью. В результате выполения операции

сдвига на выходах кодера появляются выходные символы, которые и формируют кодовые слова.

Сверточный кодер представлен на Рис. 3.7.

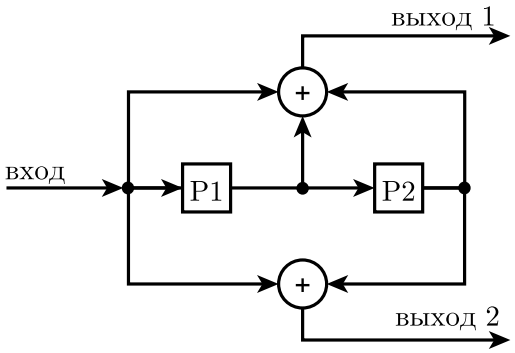

Рис. 3.7 — Сверточный кодер

3.3.2 Алгоритм Витерби

Классическим методом коррекции ошибок, по праву считавшимся лучшим в течение нескольких десятилетий, является декодер Витерби, применяемый для декодирования сверточных кодов. Данный алгоритм является оптимальным и легко реализуемым для коротких сверточных кодов. В связи с этим сверточные коды, декодируемые с помощью алгоритма Витерби, применяются в большинстве стандартов систем передачи данных, например, в беспроводных сетях стандартов IEEE 802.11, IEEE 802.16, дальней космической связи CCSDS, спутниковой связи TIA-1008 и др. Использование длинных (с конструктивной длиной более 9) и потенциально более эффективных кодов в декодере Витерби представляется нецелесообразным из-за экспоненциального роста сложности его реализации от длины кода. Поэтому долгое время усилия многих специалистов были направлены на разработку алгоритмов декодирования, которые способны эффективно декодировать длинные коды при небольшой сложности реализации.

В основе алгоритм Витерби лежит принцип максимума правдоподобия. В процессе декодирования кодового слова декодер Витерби находит кодовую последовательность, ближайшую к принятой.

а) Вычисление метрик рёбер решётки. Метрика ребра решётки на вычисляется как Хэммингове расстояние между меткой ребра и двоичной последовательностью на входе декодера. Расстояние между последовательностями по Хэммингу — количество различающихся битов в последовательностях.

б) Прибавить, сравнить, выбрать. Для каждого состояния и соответствующей пары рёбер, входящих в данное состояние из двух возможных предшествующих, вычислить и сравнить следующие суммы. Выбрать ребро с меньшей суммой.

в) Декодирование символов. Вычислить декодируемый символ и записать его в буфер накопленной последовательности. При этом выдать ранее сохраненный символ на выход декодера.

Особенностью Декодер Витерби является то, что при последовательном декодировании при возникновении ошибок в канале метрики путей растут. Для устранения этого недостатка выполняется процедура нормализации метрик. Существует два способа нормализации метрик: пороговый и модулярный.

Пример декодирования кодового слова с использованием алгоритма Витерби приведен на на Рис. 3.8.

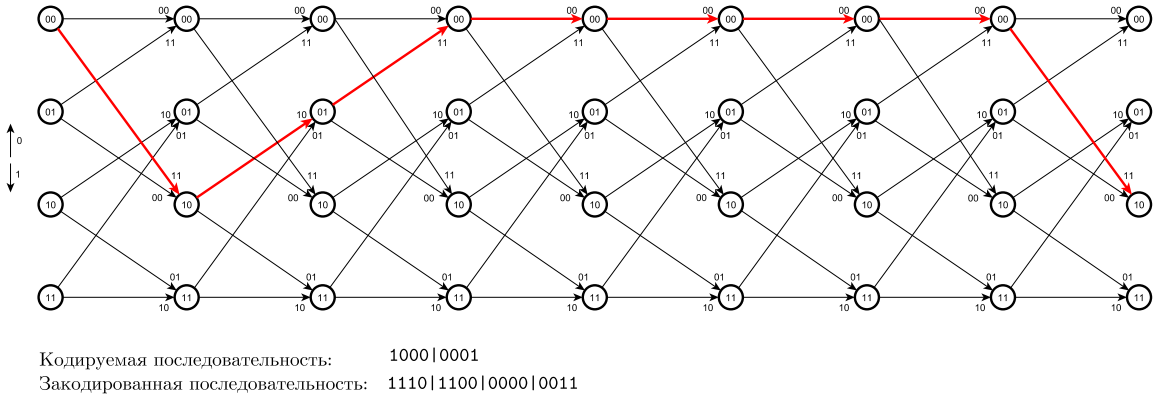

Рис. 3.8 — Декодирование кодового слова с использованием алгоритма Витерби

3.4 Каскадные коды

Каскадные коды используются в практике передачи дискретных сигналов в качестве методов реализации кодов большой длины и высокой корректирующей способности. Эта цель может быть достигнута применением нескольких ступеней кодирования; наибольшее распространение получили две ступени кодирования различными кодами, например по схеме, показанной на Рис. 3.9.

Поступающие от источника сообщений данные разбиваются на блоки из недвоичных (m-ичных) символов, содержащие k информационных элементов ($k = km$), которые кодируются недвоичным (n, k)кодом. Очевидно, что каждый коэффициент (n, k) кода содержит m двоичных элементов. Последовательность (n, k) кода, состоящая из $n_1 = nm$ двоичных элементов поступает на внутренний кодер, где разбивается на m-элементные блоки, которые кодируются внутренним (n_2, m) кодом. Число кодовых комбинаций $N = K + R$, поэтому на выходе внутреннего кодера число двоичных элементов будет $n = n_2 * N$.

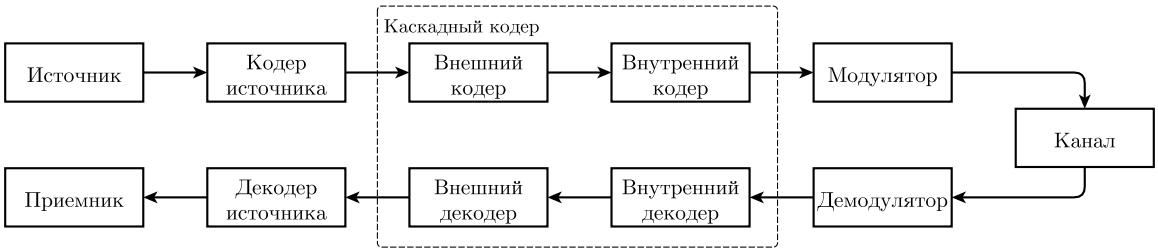

Рис. 3.9 — Система передачи информации при использовании каскадного кодирования

3.4.1 Получение каскадных кодов

Схема системы передачи информации, используемая для анализа кодов в данной работе представлена на Рис. 3.10.

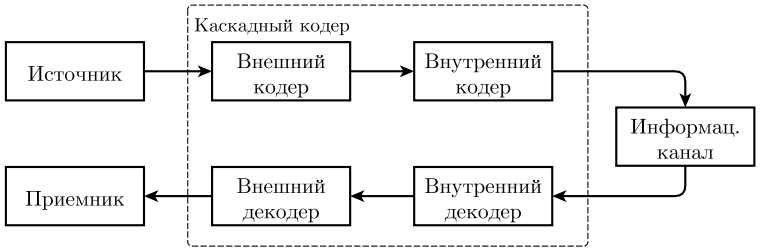

Рис. 3.10 — Система передачи информации для анализа кодов

В данной работе внешним кодером выступает БЧХ-кодер, который генерирует кодовое слово БЧХ-кода с параметрами $(15, 5, 7)$. Установлено, что данный код гарантированно исправляет ошибки в количестве $t \leq 3$. Указанное слово поступает на вход внутреннего кодера, которым является сверточный кодер. Он принимает кодовое слово БЧХ -кода и кодирует его в слово сверточного кода с параметрами $(2, 1)$. На выходе получаем кодовые слова сверточного кода, которые поступают в модулятор. В данной работе модулятор не используется по причине акцента внимания на анализ кодов. Таким образом, кодовые слова сверточного кода поступают в информационный канал, где происходит их искажение.

3.4.2 Декодирование каскадных кодов

При декодировании каскадного кода выполняется ряд обратных кодированию процедур. Так, при демодуляции на удаленной стороне имеем кодовое слово сверточного кода. Данное слово декодируется сверточным декодером, который построен на основе алгоритма Витерби. В результате декодирования нескольких кодовых слов имеем кодовое слово БЧХ-кода, которое подается на вход БЧХ-декодеру. В результате декодирования данного слова имеем исходные данные, то есть данные сгенерированные кодером источника. Очевидно, что принцип каскадного декодирова-

ния аналогичен действию получателя искаженной телеграммы. Так, если искажены буквы в словах (независимые ошибки в канале), то они могут быть обнаружены и исправлены с помощью других букв того же слова (внутреннее кодирование решает эту задачу), а если же отдельные слова искажены до неузнаваемости (группирующиеся ошибки пакеты), то обнаруженное наличие таких ошибок и тем более их исправление возможно только с помощью других слов предложения или текста в целом (внешнее кодирование). В рассмотренной аналогии основу внутреннего кода составляет избыточность за счет связей между буквами в словах, а внешнего кода за счет связей слов в предложении.

4 Описание программной модели

4.1 Общие сведения

Программная модель представляет собой консольное приложение, предназначенное для моделирования и анализа системы передачи данных при использовании каскадных кодов. Приложение имеет интерфейс командной строки и позволяет выполнять конфигурацию модели перед ее запуском путем указания параметров используемых кодов. В случае, если данные параметры не указываются, модель конфигурируется параметрами по умолчанию. Полный набор опций разработанной модели представлен в Листинге 4.1.

```
Administrator@DNAPC /cygdrive/d/SUAI/8sem/Alexeev/msvs10/scm_new/
   csm/build/bin/Release/app
$ ./app --h
Usage: csm [-uv] [-g <int>] [-l <int>] [-e <int>] [-d <int>] [-b <
   double>] [-p <string>] <file> [<file>]... [--help] [--version]
Program performs simulation of data transfer in communication
   system. The system is structured in the following manner:
[Transmitter]->[BCH encoder]->[Convolutional Encoder]->[Binary
   Symmetric Channel]->[Viterbi Decoder]->[BCH decoder]->[Receiver]
The following options are available:
  -g, --galois=<int>        define Evariste Galois field degree (
     between 2..20, default is 4)
  -l, --length=<int>        define BCH code length (default is 15)
  -e, --errors=<int>        define BCH code error correction
     property (default is 3)
  -d, --dbuffer=<int>       define internal buffer size of trellis
     node in Viterbi Decoder (default is 2 frames)
  -b, --ber=<double>        define channel Bit Error Rate (default
     is 1e-12)
  -p, --postfix=<string>    define out file postfix (default is '
     transferred')
  <file>                    define input file(s)
  -u, --gui                 run under GUI
  -v, --verbose, --debug    verbose messages
  --help                    print this help and exit
  --version                 print version information and exit
```

Листинг 4.1 — Вывод информации о доступных опциях

В качестве исходного файла, предназначенного для его использования при моделировании процесса передачи данных, указывается потенциально любой файл. Следует учесть, что в случае, если исходный файл имеет размер, больший чем 1МБ,

то процесс моделирования затянется на достаточное время. Это связано с тем, что при реализации данной модели акцент на оптимизацию по времени выполнения процеса симуляции, размеру потребляемой памяти приложением, оценке производительности приложения в целом, не делался. Главной задачей было убедиться в работоспособности модели для получения с нее результатов моделирования.

4.2 Используемый инструментарий и техника разработки

При построении модели системы передачи информации использовалась интегрированная среда разработки Microsoft Visual Studio 2010. Разработка велась преимущественно на языке высокого уровня C с добавлением конструкций, характерных для языка C++. При построении модели применялось компонентно-ориентированное программирование — парадигма программирования, ключевой фигурой которой является компонент. В качестве такого компонента выступает библиотека, реализующая необходимый функционал, и обладающая своим интерфейсом. Несмотря на широкое использование данного подхода при построении систем на C++, указанный подход являлся основой для построения системы в целом. В результате изменения в существующую систему вносятся путём создания новых компонентов в дополнение или в качестве замены ранее существующих. Особое значение было уделено интерфейсам компонентов и технике, позволяющей строить гибкие системы и использовать большинство широко распространенных паттернов проектирования в случае использования языка C.

При реализации модели автор старался применять разработку через тестирование (test-driven development, TDD) и модульное тестирование — техники разработки программного обеспечения, которые основываются на повторении очень коротких циклов разработки. Ключевыми моментами являются частые релизы, рефакторинг, контроль архитектуры.

4.3 Архитектура модели

Модель реализуется в виде набора компонентов, представленных в виде статических библиотек, каждая из которых решает свою, четко обозначенную задачу. Так, в проекте используются библиотеки, предназначенные для кодирования/декодирования БЧХ-кодов, кодирования/декодирования сверточных кодов. Характерной особенностью является разнесение ответственности компонентов и разбиение кодеров и декодеров на кодеры переднего плана и кодеры заднего плана. Такая практика позволяет контролировать архитектуру отдельных компонентов библиотеки и гибко управлять интерфейсом компонентов. Более того, практика дает возможность гибкого модифицирования или же замены алгоритма работы компонента заднего плана.

Помимо этого, имеются библиотеки, служащие для моделирования канала передачи данных, тестового окружения.

5 Запуск программной модели

Генерируемая инормация после после окончания моделирования представлена в Листинге 5.1. Характерной особенностью является автоматическое тестирвоание на предмет корректности принятых данных на удаленной стороне.

```
Administrator@DNAPC /cygdrive/d/SUAI/8sem/Alexeev/msvs10/scm_new/
   TeX/inc/img/chapter_4
$ ./app -b 0.0001 img_test_01.gif
Performing 'img_test_01.gif'
Progress: 100%

... transmitter stopped
... transmitter settings ...
data size: 156456
frame size: 5
... transmitter statistics ...
transmitted frames: 31292
transmitted bits: 156460

... receiver stopped
... receiver settings ...
data size: 156456
frame size: 5
... receiver statistics ...
received frames: 31292
valid frames: 31292 (100.00%)
invalid frames: 0 (0.00%)
received bits: 156460
valid bits: 156460 (100.00%)
invalid bits: 0 (0.00%)
BER: 0.000000
FER: 0.000000

... bch encoder stopped
... bch encoder settings ...
galois field degree: 4
code length: 15
information symbols quantity: 5
error correction property: 3
... bch encoder statistics ...
encoded frames: 31292
```

```
generated codewords: 31292
transmitted bits: 469380

... bch decoder stopped
... bch decoder settings ...
galois field degree: 4
code length: 15
information symbols quantity: 5
error correction property: 3
... bch decoder statistics ...
received bits: 469380
received bits (valid): 469380
received bits (corrupted): 0
received codewords: 31292
received codewords (valid): 31292
received codewords (corrupted): 0
generated frames: 31292

... cnv encoder stopped
... cnv encoder settings ...
quantity of registers: 2
codeword length: 30
... cnv encoder statistics ...
encoded frames: 31292
generated codewords: 31292
transmitted bits: 938760

... cnv decoder stopped
... cnv decoder settings ...
decoded sequence buffer size: 31
... cnv decoder statistics ...
delayed frames: 2
received bits: 938820
received bits (valid): 935880
received bits (corrupted): 98
received codewords: 31294
received codewords (valid): 31196
received codewords (corrupted): 98
generated frames: 31294

... channel bs stopped
```

```
77 ... channel bs settings ...
78 BER: 0.000100
79 ... channel bs statistics ...
80 bits transferred: 938820
81 bits corrupted: 98 (0.01%)
82 test: OK
```

Листинг 5.1 — Вывод информации после окончания моделирования

6 Результаты моделирования

Данный раздел содержит результаты моделирвоания передачи данных при использовании каскадного кодирования. Моделирование выполнялось с использованием разработанного и протестированного приложения. Основная цель моделирования — получение наглядных результатов, демонстрирующих эффективность каскадного кодирования и корреллирующих с теоретическими ожиданиями, а также оценка частоты появления ошибок на удаленной стороне.

6.1 Передача изображения по системе моделирования

На Рис. 6.1 – 6.12 представлены исходное изображение, моделирование передачи которого производилось на программной модели. На вход поступают данные изображения, затем они искажаются в канале в зависимости от текущих параметров канала. Далее искаженные данные принимаются на удаленной стороне и декодируются. Под каждым изображением указаны параметры модели. Нотация параметров коррелирует с обозначением ключей приложения, реализующего программную модель.

Из данных рисунков видно, что приемлемое качество изображения сохраняется даже при $BER = 0.017011$, что говорит о эффективности методов каскадного кодирования. При использовании $BER = 0.020413$ изображение начинает содержать искажения, которые увеличиваются при увеличении BER соответственно. В реальной жизни, как правило, уровень ошибок в канале связи гораздо меньше указанных значений (обычно он меньше 10^{-5}). Это означает, что принятое изображение будет в таком случае примлемого качества и пользователь на удаленной стороне практически не заметит разницу в случае, если даже при декодировании не удалось восстановить исходную информацию.

Искажения изображений связаны с тем, что изображения изначально были представлены в формате GIF. Формат способен хранить сжатые данные без потери качества в формате не более 256 цветов. Следовательно, по причине того, что данные сжаты, при их повреждении наблюдается эффект искажения части изображения. Для получения неразмытых изображений был использован формат BMP, отличающийся отсутствием кодирования данных. На Рис. 6.13 – 6.16 показаны исходное и переданные изображения.

6.2 Оценка частоты появления ошибок на удаленной стороне

На Рис. 6.17 показана зависимость BER на приемнике от BER в канале при условии, что размер буфера накопленной последовательности в узле решетки декодера Витерби составляет 1 фрейм (15 символов). На Рис. 6.18 и Рис. 6.19 показаны

Рис. 6.1 — Исходное изображение

Рис. 6.2 — g=4, l=15, e=3, d=31, b=0.000024

Рис. 6.3 — g=4, l=15, e=3, d=31, b=0.009844

Рис. 6.4 — g=4, l=15, e=3, d=31, b=0.011813

аналогичные зависимости, но при увеличении размера буфера до 2 и 10 фреймов соответственно.

На Рис. 6.20 показаны указанные зависимости совместно. Можно наблюдать, что в целом размер буфера накопленной последовательности в узле решетки декодера Витерби не сильно влияет на поведение кривой.

На Рис. 6.21 показаны указанные зависимости совместно при низких значениях BER. На этих рисунках можно аналогично наблюдать, что в целом размер буфера накопленной последовательности в узле решетки декодера Витерби не сильно влияет на поведение кривой.

На Рис. 6.22 показаны указанные зависимости в узком промежутке BER. На этих рисунках уже можно наблюдать, что размер буфера накопленной после-

Рис. 6.5 — g=4, l=15, e=3, d=31, b=0.014176

Рис. 6.6 — g=4, l=15, e=3, d=31, b=0.017011

Рис. 6.7 — g=4, l=15, e=3, d=31, b=0.020413

Рис. 6.8 — g=4, l=15, e=3, d=31, b=0.024496

довательности в узле решетки декодера Витерби определенно влияет на поведение кривой. При увеличении размера буфера уменьшается значение BER на приемнике удаленной стороны. Это происходит потому, что в данном случае декодер Витерби содержит большее количество накопленных символов для различных путей в решетке. Данные пути в этом случае с большей вероятностью начинаются в одном узле и разветвляются по прошествии большего числа шагов. Следовательно, вероятность выдачи корректного символа выше.

Для получения теоретических результатов и сравнения с ними результатов, полученных на практике был выполнен следующий эксперимент. Выполнялось моеделирование передачи данных при отключенном БЧХ-кодировании/декодировании, то есть использовалось только сверточное кодирование и декодирование по алгоритму

Рис. 6.9 — g=4, l=15, e=3, d=31, b=0.029395

Рис. 6.10 — g=4, l=15, e=3, d=31, b=0.035275

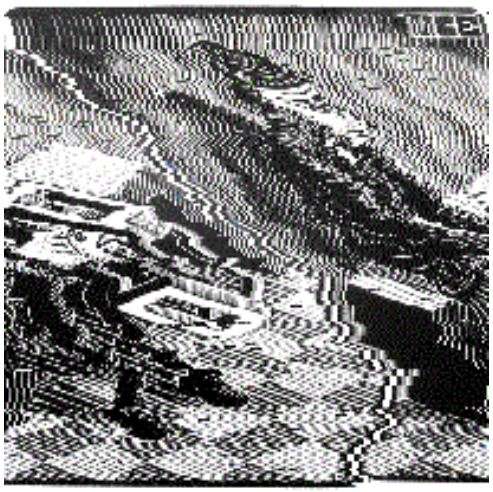

Рис. 6.11 — g=4, l=15, e=3, d=31, b=0.042329

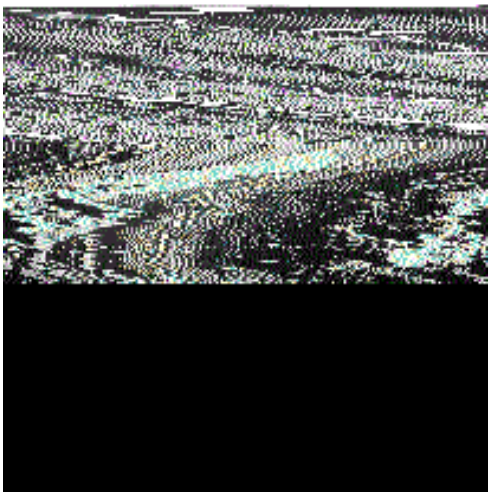

Рис. 6.12 — g=4, l=15, e=3, d=31, b=0.060954

Витерби. Данная конфигурация обусловлена тем, что достаточно просто (для использовавшегося сверточного кодера) построить верхнюю оценку зависимости BER на приемнике от BER в канале.

Указанные зависимости в логарифмическом масштабе представлены на Рис. 6.23. График черного цвета показывает кривую, получающуюся в результате теоретических вычислений. Из данного рисунка можно наблюдать, что теоретический результат (верхняя граница) близок к результатам, полу-ченным благодаря моделированию.

Данный результат полностью согласуется с теорией. Более того, по предыдущим иллюстрациям можно наблюдать, что зависимость уменьшения BER на прием-

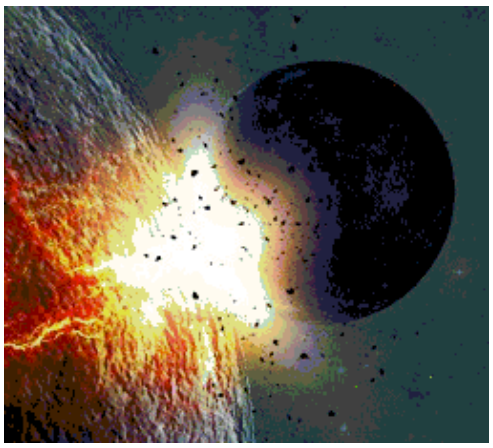

Рис. 6.13 — Исходное изображение

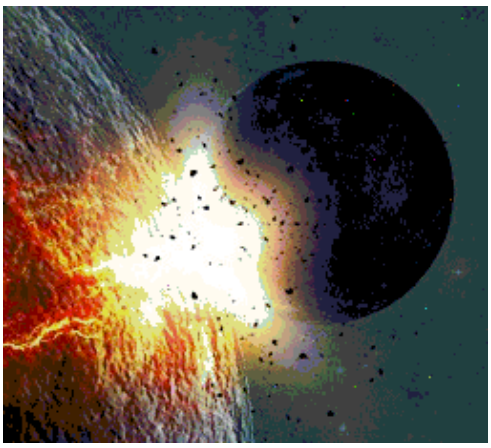

Рис. 6.14 — g=4, l=15, e=3, d=31, b=0.02

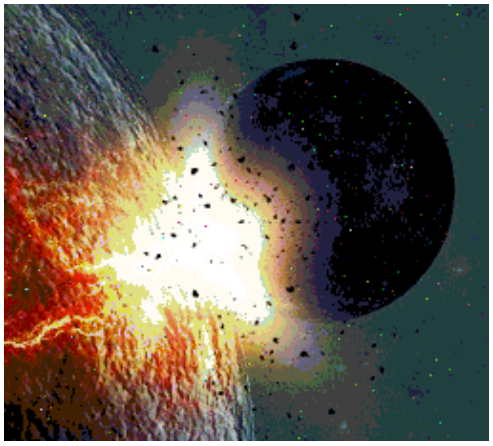

Рис. 6.15 — g=4, l=15, e=3, d=31, b=0.04

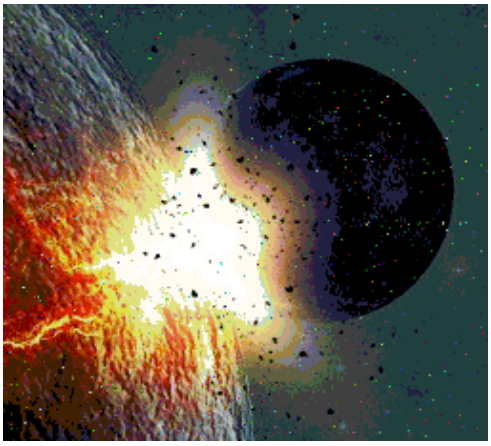

Рис. 6.16 — g=4, l=15, e=3, d=31, b=0.05

нике от применения буфера накопленной последовательности в узле решетки декодера Витерби большего размера сохраняется.

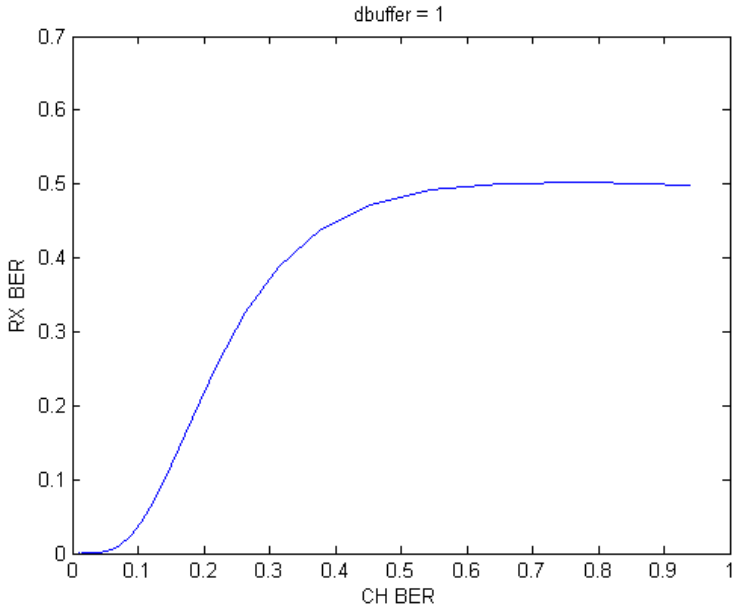

Рис. 6.17 — Зависимость BER на приемнике от BER в канале, dbuffer=1

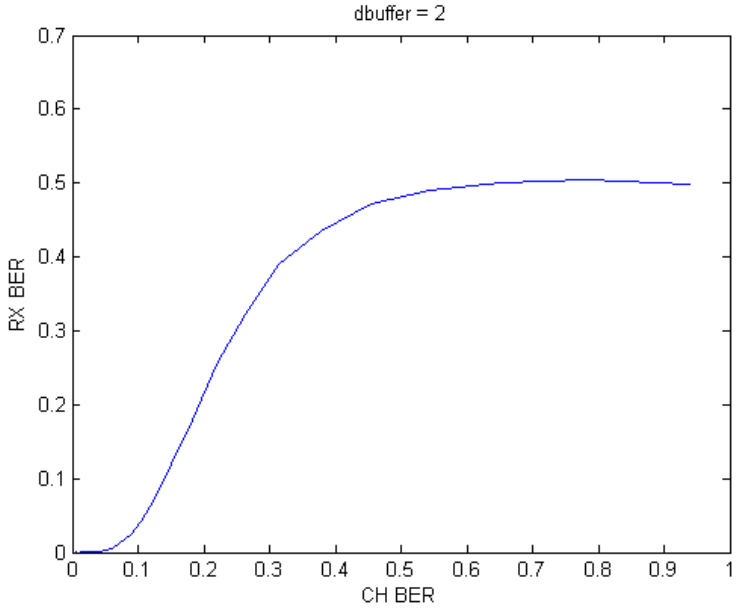

Рис. 6.18 — Зависимость BER на приемнике от BER в канале, dbuffer=2

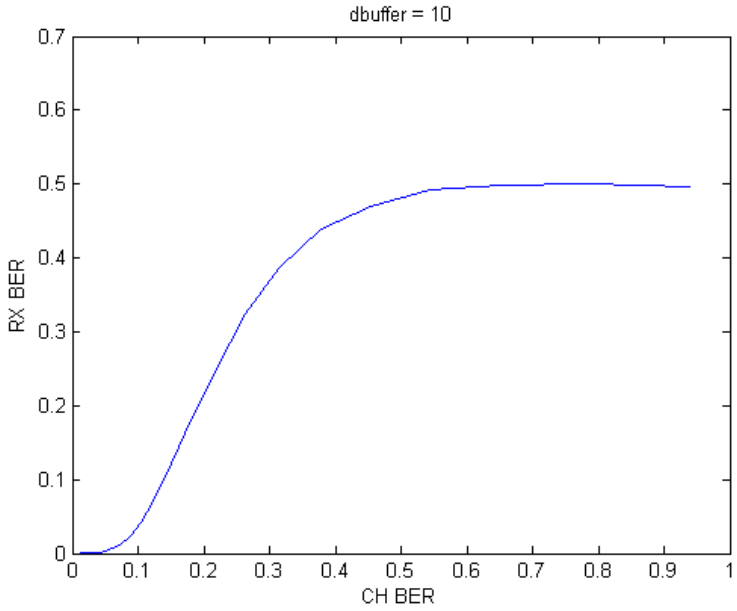

Рис. 6.19 — Зависимость BER на приемнике от BER в канале, dbuffer=10

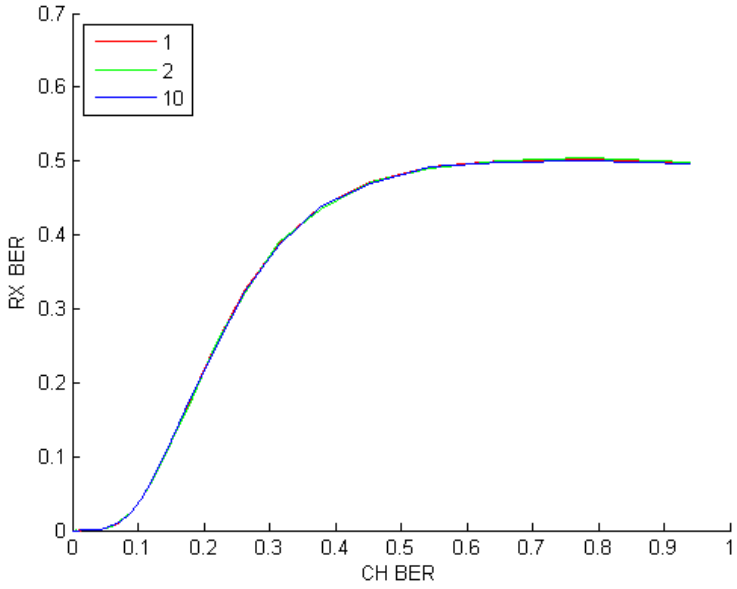

Рис. 6.20 — Зависимость BER на приемнике от BER в канале (все случаи)

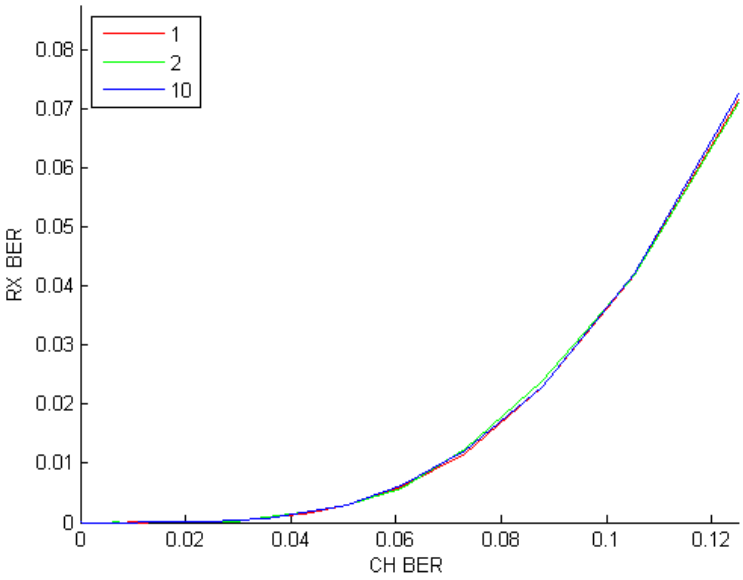

Рис. 6.21 — Зависимость BER на приемнике от BER в канале (все случаи)

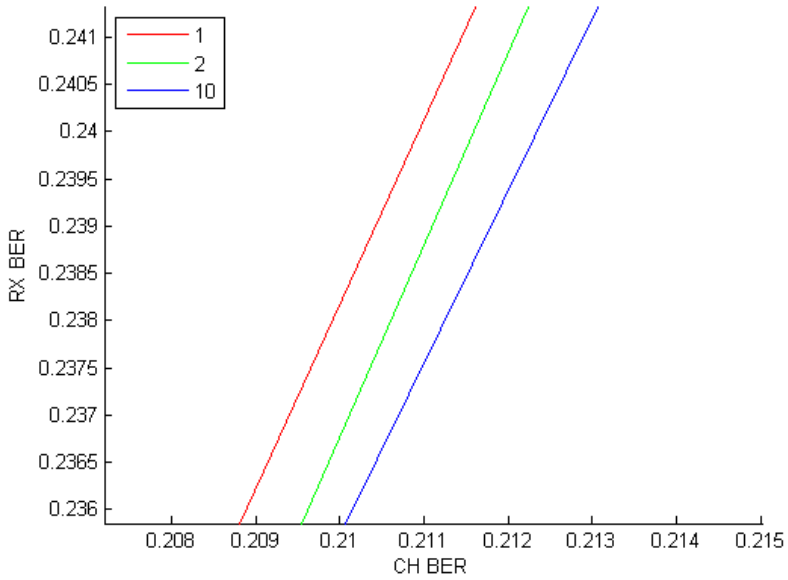

Рис. 6.22 — Зависимость BER на приемнике от BER в канале (все случаи)

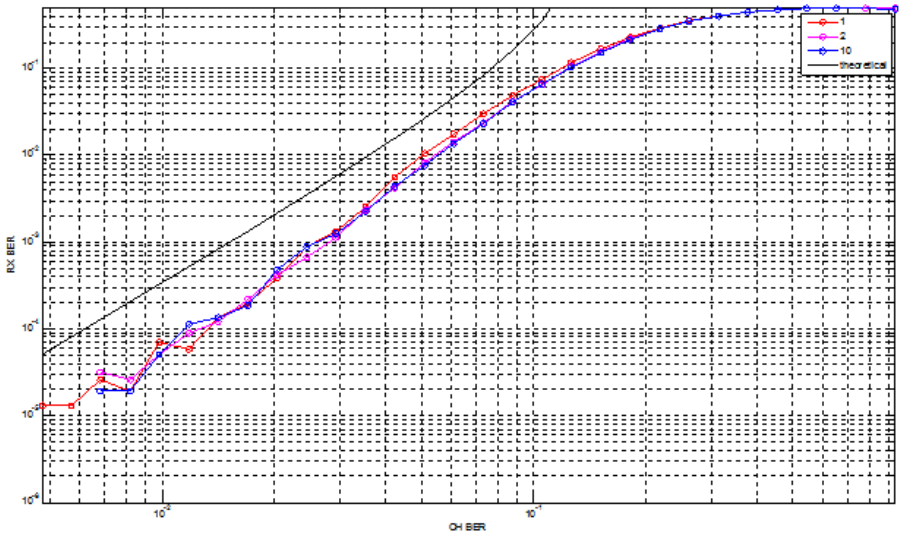

Рис. 6.23 — Верхняя граница и полученные на практике результаты (БЧХ-кодирование/декодирование не используется)

Заключение

Представленная работа посвящена реализации и исследованию модели системы передачи информации при использовании каскадных кодов (коды БЧХ и сверточные коды). Указанная модель системы была реализована в виде набора независимых компонентов, каждый из которых был тщательно отлажен. По завершению этапа построения и реализации модели она была использована для имитационного моделирования процесса передачи данных изображения по системе для наглядной демонстрации результатов передачи данных и эффективности каскадного кодирования и для оценки зависимости BER на приемнике от BER в канале. Выполнялось имитационное моделирование при отключенном механизме БЧХ- кодирования/декодирования для получения теоретических результатов и сравнения с ними результатов, полученных на практике в предыдущем случае.

Были получены результаты, которые показали, что сверточное кодирование и использование алгоритма Витерби при декодировании позволяют получать данные (изображения) приемлемого качества даже при значении BER в канале, равном $BER = 0.017011$. По сравнению с аналогичным значением в реальных физических каналах (не более 10^{-5}) данный результат говорит о эффективности применения сверточного кодирования.

Использование БЧХ-кодирования позволяет восстанавливать информацию, которую не смог восстановить декодер Витерби. В реальных же системах, внешний декодер исправляет пакеты ошибок, которые генерирует декодер Витерби, в случае, если он не в состоянии исправить их. Таким образом совместное использование указанных способов кодирования позволяет повысить надежность передачи данных.

Другой результат связан с определением оптимального размера буфера накопленной последовательности узла решетки, которая применяется в декодере, работающем по алгоритму Витерби. Экспериментальные исследования показали, что действительно, увеличение размера указанного буфера позволяет получить выигрыш. Однако, чем больше буфер, тем меньше заметен выигрыш и при разработке реального устройства необходимо учитывать экономическую составляющую. Например, стоимость памяти, необходимой для реализации декодера Витерби при построении реальной системы передачи информации.

В целом, полученные результаты полностью соответствуют теории, что говорит о корректности реализации исследованных способов кодирования.

Список использованных источников

1. *Морелос-Сарагоса, Р.* Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение / Р. Морелос-Сарагоса. — Техносфера, 2006.

2. *Кларк Дж. мл., Кейн Дж.* Кодирование с исправлением ошибок в системах цифровой связи / Кейн Дж. Кларк Дж., мл. — Радио и связь, 1987.

3. *Витерби А.Д., Омура Дж.К.* Принципы цифровой связи и кодирования / Омура Дж.К. Витерби А.Д. — Радио и связь, 1982.

4. *Э., Берлекэмп.* Алгебраическая теория кодирования / Берлекэмп Э. — Мир, 1971.

5. *Р., Галлагер.* Теория информации и надежная связь / Галлагер Р. — Советское радио, 1974.

6. *Мак-Вильямс Ф.Дж., Слоэн Н.Дж.А.* Теория кодов, исправляющих ошибки / Слоэн Н.Дж.А. Мак-Вильямс Ф.Дж. — Связь, 1979.