

Санкт-Петербургский политехнический университет Петра Великого
Кафедра компьютерных систем и программных технологий

Отчёт по лабораторной работе

Дисциплина: Телекоммуникационные технологии

Тема: Помехоустойчивое кодирование.

Выполнил студент гр. 33501/3
Преподаватель

Вотчицев К. В.
Богач Н.В.

Санкт-Петербург
2018 г.

0 Содержание

1	Цель работы	2
2	Постановка задачи	2
3	Теоретический раздел	2
3.1	Классификация похемоустойчивых кодов	2
4	Ход работы	3
4.1	Код Хэмминга	3
4.1.1	Кодирование с помощью встроенных функций encode/decode	3
4.1.2	Кодирование с помощью проверочной и генераторной матрицы с вычислением синдром	4
4.2	Циклический код	6
4.3	Коды БЧХ	9
4.4	Коды Рида-Соломона	10
5	Выводы	11

1 Цель работы

Изучение методов помехоустойчивого кодирования и сравнение их свойств.

2 Постановка задачи

1. Провести кодирование/декодирование сигнала, полученного с помощью функции `randerr` кодом Хэмминга 2-мя способами: с помощью встроенных функций `encode/decode`, а также через создание проверочной и генераторной матриц и вычисление синдрома. Оценить корректирующую способность кода.
2. Выполнить кодирование/декодирование циклическим кодом, кодом БЧХ, кодом Рида-Соломона. Оценить корректирующую способность кода.

3 Теоретический раздел

Коды, исправляющие ошибки, были придуманы для исправления ошибок в каналах связи с шумом. Основным средством обеспечения высокой помехоустойчивости является введение избыточности, необходимой для обнаружения и исправления ошибок. Теория кодов, исправляющих ошибки, служит теоретической базой для эффективного использования вводимой избыточности. В данной работе рассматривается проблема помехоустойчивого кодирования

3.1 Классификация помехоустойчивых кодов

Код Хемминга – это блочный код, позволяющий исправлять одиночные и фиксировать двойные ошибки. Идея кодов Хемминга заключается в разбиении данных на блоки фиксированной длины и вводе в эти блоки контрольных бит, дополняющих до четности несколько пересекающихся групп, охватывающих все биты блока.

Циклический коды являются подклассом линейных кодов, и в силу жесткой математической структуры поиск хороших кодов, исправляющих ошибки, в классе циклических кодов оказался наиболее успешным. В качестве математического аппарата для циклических кодов используются поля Галуа и классы вычетов многочленов.

Коды Боуза — Чоудхури — Хоквингема (БЧХ-коды) - в теории кодирования это широкий класс циклических кодов, представляющие собой обобщенные коды Хемминга, позволяющие исправлять кратные ошибки.

Код Рида — Соломона - частный случай БЧХ-кода, позволяющий исправлять ошибки в блоках данных.

4 Ход работы

4.1 Код Хэмминга

4.1.1 Кодирование с помощью встроенных функций encode/decode

Был написан код на языке MATLAB, проводящий кодирование и декодирование сигнала с помощью функций encode и decode

```
1 - N = 7; % Длина кодового слова
2 - K = 4; % Длина сообщения
3 - message = randerr(1,K,3) % Посылка
4 - code = encode(message,N,K); %Кодирование
5 - code(2) = not(code(2)); %Преднамеренная ошибка во 2 символе
6 - dec = decode(code,N,K) %Декодирование
7 -
```

Рис. 4.1: Код на языке Matlab

Несмотря на то, была преднамеренно допущена ошибка во 2 символе, код был успешно декодирован.

```
message =
      0      1      1      1

dec =
      0      1      1      1
```

Рис. 4.2: Результат выполнения программы

Попробуем добавить еще одну преднамеренную ошибку в другом символе.

```

1 - N = 7; % Длина кодового слова
2 - K = 4; % Длина сообщения
3 - message = randerr(1,K,3) % Посылка
4 - code = encode(message,N,K); %Кодирование
5 - code(2) = not(code(2)); %Преднамеренная ошибка во 2 символе
6 - code(4) = not(code(4)); %Преднамеренная ошибка во 4 символе
7 - dec=decode(code,N,K) %Декодирование
8
9

```

Рис. 4.3: Изменённая программа в MATLAB

В результате выполнения программы декодированное сообщение не совпало с исходным

```

message =

     1     0     1     1

dec =

     0     0     1     1

```

Рис. 4.4: Результат выполнения программы

Это связано с тем, что корректирующая способность кода равна 1.

4.1.2 Кодирование с помощью проверочной и генераторной матрицы с вычислением синдром

Был написан код на языке MATLAB, проводящий кодирование и декодирование сигнала через создание проверочной и генераторной матриц и вычисления синдрома

```

1 - N = 7; % Длина кодового слова
2 - K=4; %Длина сообщения
3 - message = randerr(1,K,3)
4 - [h,g] = hammgcn(3); % Генерация проверочной и порождающей матриц для кода Хэмминга
5 - mes=message*g;
6 - mes=rem(mes,ones(1,N).*2);
7 - mes(2)=not(mes(2)); %Преднамеренная ошибка в 2 символе
8 - syndrom=mes*h';
9 - syndrom=rem(syndrom,ones(1,N-K).*2)
10 - t1 = syndtable(h) %Декодированная таблица
11 - t2 = bi2de(syndrom,'left-msb') % Преобразование вектора в неотрицательное число, где первый столбец - старший разряд
12 - correction_vector = t1(t2+1,:) %Корректирующий вектор
13 - correction_code = rem(correction_vector+mes,2)

```

Рис. 4.5: Код на языке MATLAB

Код был успешно декодирован

```
message =  
  
    1    1    1    0  
  
syndrom =  
  
    0    1    0  
  
t1 =  
  
    0    0    0    0    0    0    0  
    0    0    1    0    0    0    0  
    0    1    0    0    0    0    0  
    0    0    0    0    1    0    0  
    1    0    0    0    0    0    0  
    0    0    0    0    0    0    1  
    0    0    0    1    0    0    0  
    0    0    0    0    0    1    0  
  
t2 =  
  
    2  
  
correction_vector =  
  
    0    1    0    0    0    0    0  
  
correction_code =  
  
    0    1    0    1    1    1    0
```

Рис. 4.6: Результат выполнения программы

Попробуем добавить еще одну преднамеренную ошибку в другом символе.

```
1 - N = 7; % Длина кодового слова  
2 - K=4; %Длина сообщения  
3 - message = randerr(1,K,3)  
4 - [h,g] = hamngen(3); % Генерация проверочной и порождающей матриц для кода Хэмминга  
5 - mes=message*g;  
6 - mes=rem(mes,ones(1,N).*2);  
7 - mes(2)=not(mes(2)); %Преднамеренная ошибка в 2 символе  
8 - mes(4)=not(mes(4)); %Преднамеренная ошибка в 4 символе  
9 - syndrom=mes*h';  
10 - syndrom=rem(syndrom,ones(1,N-K).*2)  
11 - t1 = syndtable(h) %Декодированная таблица  
12 - t2 = bi2de(syndrom,'left-msb') % Преобразование вектора в неотрицательное число, где первый столбец - старший разряд  
13 - correction_vector = t1(t2+1,:) %Корректирующий вектор  
14 - correction_code = rem(correction_vector+mes,2)
```

Рис. 4.7: Изменённая программа в MATLAB

В результате выполнения программы декодированное сообщение не совпало с исходным

```

message =

    1    1    0    1

syndrom =

    1    0    0

t1 =

    0    0    0    0    0    0    0
    0    0    1    0    0    0    0
    0    1    0    0    0    0    0
    0    0    0    0    1    0    0
    1    0    0    0    0    0    0
    0    0    0    0    0    0    1
    0    0    0    1    0    0    0
    0    0    0    0    0    1    0

t2 =

    4

correction_vector =

    1    0    0    0    0    0    0

correction_code =

    1    1    0    0    1    0    1

```

Рис. 4.8: Результат выполнения программы

Как было написано ранее, это связано с тем, что корректирующая способность кода равна 1.

4.2 Циклический код

Был написан код на языке MATLAB, проводящий кодирование и декодирование сигнала с помощью циклического кода

```

1 - N = 7; %Длина кодового слова
2 - K = 4; %Длина сообщения
3 - message = randerr(1,K,3)
4 - polynome = cyclpoly(N,K); %Генерация полинома для циклического кода (X^3+X+1)
5 - [h,g] = cyclgen(N,polynome); % Проверка на четность и генерация матрицы для циклического кода
6 - mes=message*g;
7 - mes=rem(mes,ones(1,N).*2);
8 - mes(2)=not(mes(2)); %Преднамеренная ошибка в 2 символе
9 - syndrom=mes*h';
10 - syndrom=rem(syndrom,ones(1,N-K).*2)
11 - t1 = syndtable(h) %Декодированная таблица
12 - t2 = bi2de(syndrom,'left-msb') % Преобразование вектора в неотрицательное число, где первый столбец - старший разряд
13 - correction_vector = t1(t2+1,:) %Корректирующий вектор
14 - correction_code = rem(correction_vector+mes,2)

```

Рис. 4.9: Код на языке MATLAB

Код был успешно декодирован

```

message =

     0     1     1     1

syndrom =

     0     1     0

t1 =

     0     0     0     0     0     0     0
     0     0     1     0     0     0     0
     0     1     0     0     0     0     0
     0     0     0     0     0     0     1
     1     0     0     0     0     0     0
     0     0     0     1     0     0     0
     0     0     0     0     0     1     0
     0     0     0     0     1     0     0

t2 =

     2

correction_vector =

     0     1     0     0     0     0     0

correction_code =

     0     1     0     0     1     1     1

```

Рис. 4.10: Результат выполнения программы

Добавим еще одну ошибку

```

1 - N = 7; %Длина кодового слова
2 - K = 4; %Длина сообщения
3 - message = randerr(1,K,3)
4 - polynome = cyclpoly(N,K); %Генерация полинома для циклического кода (X^3+X+1)
5 - [h,g] = cyclgen(N,polynome); % Проверка на четность и генерация матрицы для циклического кода
6 - mes=message*g;
7 - mes=rem(mes,ones(1,N).*2);
8 - mes(2)=not(mes(2)); %Преднамеренная ошибка в 2 символе
9 - mes(1)=not(mes(1)); %Преднамеренная ошибка в 1 символе
10 - syndrom=mes*h';
11 - syndrom=rem(syndrom,ones(1,N-K).*2)
12 - t1 = syndtable(h) %Декодированная таблица
13 - t2 = bi2de(syndrom,'left-msb') % Преобразование вектора в неотрицательное число, где первый столбец - старший разряд
14 - correction_vector = t1(t2+1,:) %Корректирующий вектор
15 - correction_code = rem(correction_vector+mes,2)

```

Рис. 4.11: Изменённая программа в MATLAB

В результате выполнения программы декодированное сообщение не совпало с исходным

```

message =

     1     1     1     0

syndrom =

     1     1     0

t1 =

     0     0     0     0     0     0     0
     0     0     1     0     0     0     0
     0     1     0     0     0     0     0
     0     0     0     0     0     0     1
     1     0     0     0     0     0     0
     0     0     0     1     0     0     0
     0     0     0     0     0     1     0
     0     0     0     0     1     0     0

t2 =

     6

correction_vector =

     0     0     0     0     0     1     0

correction_code =

     0     1     0     1     1     0     0

```

Рис. 4.12: Результат выполнения программы

Следовательно, корректирующая способность кода равна 1.

4.3 Коды БЧХ

Был написан код на языке MATLAB, проводящий кодирование и декодирование сигнала с помощью Кода БЧХ

```
1 - N = 15; % Длина кодового слова
2 - K = 5; %Длина сообщения
3 - nwords = 10; %Кол-во слов для кодирования
4 - message = gf(randerr(nwords,K,3)) % формирования сообщения. Это массив элементов конечного поля
5 - t = bchnumerr(N,K) %Кол-во исправляемых ошибок
6 - encoded_code = bchenc(message,N,K); %Кодирование
7 - noise_code = encoded_code + randerr(nwords,N,1:t); %Добавление ошибок к словам
8 - decoded_code = bchdec(noise_code,N,K) %Декодирование
9 - if(message == decoded_code)
10 -     disp ("Successful decoding")
11 - end;
```

Рис. 4.13: Код на языке MATLAB

Код был успешно декодирован

```
message = GF(2) array.

Array elements =

     1     1     1     0     0
     1     0     0     1     1
     1     1     1     0     0
     1     1     0     0     1
     1     1     1     0     0
     0     0     1     1     1
     0     0     1     1     1
     0     0     1     1     1
     1     1     1     0     0
     0     0     1     1     1

t =

     3

decoded_code = GF(2) array.

Array elements =

     1     1     1     0     0
     1     0     0     1     1
     1     1     1     0     0
     1     1     0     0     1
     1     1     1     0     0
     0     0     1     1     1
     0     0     1     1     1
     0     0     1     1     1
     1     1     1     0     0
     0     0     1     1     1

Successful decoding
```

Рис. 4.14: Результат выполнения программы

Сообщение было успешно декодировано после добавления ошибок

Корректирующая способность кода равна 3, при длине кодового слова 15 и при длине сообщения 5

4.4 Коды Рида-Соломона

Был написан код на языке MATLAB, проводящий кодирование и декодирование сигнала с помощью кода Рида — Соломона (7,3)

```

1 - M = 3;           % Кол-во бит на символ
2 - N = 7;           % Длина кодового слова
3 - K = 3;           % Длина сообщения
4 - message = gf([1 3 7;5 1 1;4 5 2],M) %Генерация трех-битового слова
5 - code = rsenc(message,N,K) %Кодирование сообщения кодом Рида Соломона (7,3)
6 - errors = gf([0 0 0 0 3 0 0;0 5 4 0 0 0 0;0 0 1 2 4 0 0],M); % Генерация ошибок
7 - noise_code = code + errors; %Добавление ошибок к коду
8 - [decoded_code] = rsdec(noise_code,N,K) %Декодирование кода
9 -

```

Рис. 4.15: Код на языке MATLAB

```

message = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

     1     3     7
     5     1     1
     4     5     2

code = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

     1     3     7     3     5     7     1
     5     1     1     4     5     4     0
     4     5     2     1     3     6     0

decoded_code = GF(2^3) array. Primitive polynomial = D^3+D+1 (11 decimal)

Array elements =

     1     3     7
     5     1     1
     4     5     3

```

Рис. 4.16: Результат выполнения программы

Из результатов выполнения программы видно, что при использовании восьмеричного кода Рида-Соломона(7,3) корректирующая способность кода равна 2.

5 Выводы

В ходе работы были получены знания по кодированию и декодированию различных помехоустойчивых кодов в среде MATLAB. Изученные помехоустойчивые коды, широко применяют в различных сферах - Код Рида-Соломона используется в технологии RAID 6, код Хэмминга в технологии RAID 2 в типах памяти ECC, Циклические коды применяются при записи и считывании CD и DVD, HDD.