Министерство образования и науки Российской Федерации

Федеральное Государственное Автономное Образовательное Учреждение

Высшего Образования

“Национальный Исследовательский Университет ИТМО”

Факультет программной инженерии и компьютерной техники

Дисциплина “Информатика”

**Лабораторная работа №2**

**Синтез помехоустойчивого кода**

**Вариант №412934=23**

**Выполнил:**

Дорохин Сергей Константинович

Группа: Р3111

**Преподаватель:**

Малышева Татьяна Алексеевна

Санкт-Петербург, 2023

**Содержание**

[Условие заданий………………………………………………………………..2](#ooopzph5uaf9)

[Ход работы………………………………………………………………......3-11](#h951m9c7ysi2)

1. [Задание №1………………………………………………………………3](#vz0qn263pfrt)
2. [Задание №2………………………………………………………………4](#s7k6b9bkv97y)
3. [Задание №3………………………………………………………………5](#4t2rlhzefwyt)
4. [Задание №4………………………………………………………………6](#1tcfgdcg3no2)
5. [Задание №5………………………………………………………………7](#ixixkrpmiiym)
6. [Задание №6………………………………………………………………8](#u4psqiecu04k)
7. [Дополнительное задание……………………………………………9-11](#tlfyr8eblcyz)

[Заключение……………………………………………………………………11](#muqzeb221u5z)

[Список использованных источников…………………………………...…...11](#go8bpm7od593)

**Условия заданий**

1. Определить свой вариант задания с помощью номера в ISU (он же номер студенческого билета). Вариантом является комбинация 3-й и 5-й цифр. Мой номер в ISU = 409066, тогда вариант = 96.

2. На основании номера варианта задания выбрать набор из 4 полученных сообщений в виде последовательности 7-символьного кода.

3. Построить схему декодирования классического кода Хэмминга (7;4), которую представить в отчете в виде изображения.

4. Показать, исходя из выбранных вариантов сообщений (по 4 у каждого – часть №1 в варианте), имеются ли в принятом сообщении ошибки, и если имеются, то какие. Подробно прокомментировать и записать правильное сообщение.

5. На основании номера варианта задания выбрать 1 полученное сообщение в виде последовательности 11-символьного кода.

6. Построить схему декодирования классического кода Хэмминга (15;11), которую представить в отчете в виде изображения.

7. Показать, исходя из выбранного варианта сообщений (по 1 у каждого – часть №2 в варианте), имеются ли в принятом сообщении ошибки, и если имеются, то какие. Подробно прокомментировать и записать правильное сообщение.

8. Сложить номера всех 5 вариантов заданий. Умножить полученное число на 4. Принять данное число как число информационных разрядов в

передаваемом сообщении. Вычислить для данного числа минимальное число проверочных разрядов и коэффициент избыточности.

9. Дополнительное задание №1 (позволяет набрать от 86 до 100 процентов от максимального числа баллов БаРС за данную лабораторную). Написать программу на любом языке программирования, которая на вход получает набор из 7 цифр «0» и «1», записанных подряд, анализирует это сообщение на основе классического кода Хэмминга (7,4), а затем выдает правильное сообщение (только информационные биты) и указывает бит с ошибкой при его наличии

**Ход работы**

**Задание №1: №19**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 0 | 1 | 0 | 1 | 0 | 0 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |  |
|  |  |  |  |  |  |  |  | *S* |
| 1 | X |  | X |  | X |  | X |  |
| 2 |  | X | X |  |  | X | X |  |
| 4 |  |  |  | X | X | X | X |  |

*s1* = r1 i1 i2 i4 = 0 0 0 1 = 1

*s2* = r2 i1 i3 i4 = 1 0 0 1 = 0

*s3* = r3 i2 i3 i4 = 1 0 0 1 = 0

*S* = (*s3, s2, s1*) = 001 = 1 => ошибка на 1-ой позиции, то есть ошибка в r1.

*Правильное сообщение: 0001*

**Задание №2: №46**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 0 | 0 | 1 | 1 | 0 | 1 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |  |
|  |  |  |  |  |  |  |  | *S* |
| 1 | X |  | X |  | X |  | X |  |
| 2 |  | X | X |  |  | X | X |  |
| 4 |  |  |  | X | X | X | X |  |

*s1* = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 0 ⊕ 1 ⊕ 0 ⊕ 1 = 0

*s2* = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 1 = 1

*s3* = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 1 ⊕ 0 ⊕ 1 ⊕ 1 = 1

*S* = (*s3, s2, s1*) = 110 = 6 => ошибка на 6-ой позиции, то есть ошибка в i3.

*Правильное сообщение: 1001*

**Задание №3: №73**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |  |
|  |  |  |  |  |  |  |  | *S* |
| 1 | X |  | X |  | X |  | X |  |
| 2 |  | X | X |  |  | X | X |  |
| 4 |  |  |  | X | X | X | X |  |

*s1* = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 0 ⊕ 1 ⊕ 1 ⊕ 1 = 1

*s2* = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 0 ⊕ 1 = 0

*s3* = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 0 ⊕ 1 = 0

*S* = (*s3, s2, s1*) = 001 = 1 => ошибка на 1-ой позиции, то есть ошибка в r1.

*Правильное сообщение: 1101*

**Задание №4: №65**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 |
| 1 | 0 | 1 | 0 | 1 | 0 | 0 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 |  |
|  |  |  |  |  |  |  |  | *S* |
| 1 | X |  | X |  | X |  | X |  |
| 2 |  | X | X |  |  | X | X |  |
| 4 |  |  |  | X | X | X | X |  |

*s1* = r1 ⊕ i1 ⊕ i2 ⊕ i4 = 1 ⊕ 1 ⊕ 1 ⊕ 0 = 1

*s2* = r2 ⊕ i1 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 0 ⊕ 0 = 1

*s3* = r3 ⊕ i2 ⊕ i3 ⊕ i4 = 0 ⊕ 1 ⊕ 0 ⊕ 0 = 1

*S* = (*s3, s2, s1*) = 111 = 7 => ошибка на 7-ой позиции, то есть ошибка в i4.

*Правильное сообщение: 1101*

**Задание №5: №24**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 | r4 | i5 | i6 | i7 | i8 | i9 | i10 | i11 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |  |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | *S* |
| 1 | X |  | X |  | X |  | X |  | X |  | X |  | X |  | X |  |
| 2 |  | X | X |  |  | X | X |  |  | X | X |  |  | X | X |  |
| 4 |  |  |  | X | X | X | X |  |  |  |  | X | X | X | X |  |
| 8 |  |  |  |  |  |  |  | X | X | X | X | X | X | X | X |  |

*s1* = r1 ⊕ i1 ⊕ i2 ⊕ i4 ⊕ i5 ⊕ i7 ⊕ i9 ⊕ i11 = 0 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 = 1

*s2* = r2 ⊕ i1 ⊕ i3 ⊕ i4 ⊕ i6 ⊕ i7 ⊕ i10 ⊕ i11 = 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 = 0

*s3* = r3 ⊕ i2 ⊕ i3 ⊕ i4 ⊕ i8 ⊕ i9 ⊕ i10 ⊕ i11 = 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 = 0

*s4* = r4 ⊕ i5 ⊕ i6 ⊕ i7 ⊕ i8 ⊕ i9 ⊕ i10 ⊕ i11 = 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 = 0

*S = (s4, s3, s2, s1) =* 0001 = 1 => ошибка на 1-ой позиции, то есть, ошибка в r1.

*Правильное сообщение: 10010000001*

**Задание №6: (19 + 46 + 73 + 65 + 24) \* 4 = 908**

Число информационных разрядов в передаваемом сообщении: 908. Определим минимальное число контрольных разрядов: пусть будет *r* контрольных разрядов. Тогда справедливо следующее неравенство: , где *r* - число контрольных разрядов, *i* - число информационных разрядов. Всего разрядов (бит) в сообщении <= , а информационных разрядов (бит) <= . Справедливо следующее соотношение:

Если *r* = 9, то неравенство не выполняется: 256 - 19 < 908 !<= 512 - 11, пусть *r* = 10 => неравенство выполняется: 512-10 = 502 < 908 < 1013 = 1024 - 11, *следовательно, r = 10 - минимальное число контрольных разрядов.* Коэффициент избыточности = r / (r + i) = 10 / (10 + 908) 0.010893246

*Ответ: r = 10, k 0.010893246.*

**Дополнительное задание**

**Условие дополнительного задания**

Написать программу на любом языке программирования, которая на вход получает набор из 7 цифр «0» и «1», записанных подряд, анализирует это сообщение на основе классического кода Хэмминга (7,4), а затем выдает правильное сообщение (только информационные биты) и указывает бит с ошибкой при его наличии.

Листинг программы на языке программирования Java:

|  |
| --- |
| import java.util.Scanner;  public class Main {   public static boolean stringValidation(String line) {  if (line.length() != 7) {  return false;  }  for (char c : line.toCharArray()) {  if (c != '0' && c != '1') {  return false;  }  }  return true;  }   public static int[] stringToBits(String line) {  int[] bits = new int[line.length()];  for (int i = 0; i < line.length(); i++) {  bits[i] = Integer.*parseInt*(String.*valueOf*(line.charAt(i)));  }  return bits;  }   public static int[] calculatingSyndrome(int[] bits) {  int s1 = (bits[0] + bits[2] + bits[4] + bits[6]) % 2;  int s2 = (bits[1] + bits[2] + bits[5] + bits[6]) % 2;  int s3 = (bits[3] + bits[4] + bits[5] + bits[6]) % 2;  return new int[]{s1, s2, s3};  }   public static boolean hasError(int[] bitsTuple) {  return bitsTuple[0] != 0 || bitsTuple[1] != 0 || bitsTuple[2] != 0;  }   public static int calculatingIndex(int[] syndrome) {  StringBuilder reversedSyndrome = new StringBuilder();  for (int i = syndrome.length - 1; i >= 0; i--) {  reversedSyndrome.append(syndrome[i]);  }  return Integer.*parseInt*(reversedSyndrome.toString(), 2);  }   public static String calculatingWrongElement(int index) {  String[] symbols = {"r1", "r2", "i1", "r3", "i2", "i3", "i4"};  return symbols[index - 1];  }   public static String makeMessageFromList(int[] bits) {  bits = new int[]{bits[2], bits[4], bits[5], bits[6]};  StringBuilder message = new StringBuilder();  for (int bit : bits) {  message.append(bit);  }  return message.toString();  }   public static int getInverse(int element) {  return (element == 0) ? 1 : 0;  }   public static int[] getCorrectList(int[] bits, String wrongElement) {  int wrongElementIndex = Character.*getNumericValue*(wrongElement.charAt(wrongElement.length() - 1)) - 1 + 3;  bits[wrongElementIndex] = *getInverse*(bits[wrongElementIndex]);  return bits;  }   public static String returnCorrectMessage(int[] bits, String wrongElement) {  if (wrongElement.charAt(0) == 'r') {  return *makeMessageFromList*(bits);  }  return *makeMessageFromList*(*getCorrectList*(bits, wrongElement));  }   public static void main(String[] args) {  Scanner scanner = new Scanner(System.*in*);   System.*out*.println("Введите набор из 7 цифр \"0\" и \"1\":");  String inputString = scanner.nextLine();   if (*stringValidation*(inputString)) {  int[] bits = *stringToBits*(inputString);  int[] syndrome = *calculatingSyndrome*(bits);   if (*hasError*(syndrome)) {  int index = *calculatingIndex*(syndrome);  String wrongElement = *calculatingWrongElement*(index);  System.*out*.printf("В сообщении была допущена ошибка! Ошибка была в символе %s. Правильное сообщение: %s%n",  wrongElement, *returnCorrectMessage*(bits, wrongElement));  } else {  System.*out*.printf("В сообщении не было допущено ошибок! Правильное сообщение: %s%n",  *makeMessageFromList*(bits));  }   } else {  System.*out*.println("Некорректный ввод! Строка должна быть длины 7 и содержать только \"0\" и \"1\"!");  }  } }  **Заключение** |

В ходе этой лабораторной работы я познакомился с кодом Хэмминга и научился его декодировать, написал код на языке java, которая анализирует это сообщение на основе классического кода Хэмминга(7;4)

**Список использованных источник**

1. Вернер М.: Основы кодирования. Учебник для ВУЗов. Москва: 2004 - (Техносфера)
2. Морелос-Сарагоса Р.: Искусство помехоустойчивого кодирования. Методы, алгоритмы, применение: учеб. пособие для вузов. Москва: 2005 - (Техносфера)