Федеральное государственное автономное образовательное учреждение

высшего образования «Национальный исследовательский университет

ИТМО»

*Факультет программной инженерии и компьютерной техники*

**Лабораторная работа №2**

**По дисциплине “Информатика”**

Вариант: 91

Выполнил:

Саранча Павел

Группа: Р3109

Преподаватель:

Рудникова Тамара Владимировна

Санкт-Петербург, 2023г

Оглавление

[Задание: 2](#_Toc148303555)

[1. Вариант (91): 3](#_Toc148303556)

[2-3. Схема кода Хэмминга(7;4) 3](#_Toc148303557)

[4.Проверка кода Хэмминга на ошибки 4](#_Toc148303558)

[Задание 1 (75) 4](#_Toc148303559)

[Задание 2 (5) 4](#_Toc148303560)

[Задание 3 (47) 5](#_Toc148303561)

[Задание 4 (34) 5](#_Toc148303562)

[5 – 6. Схема кода Хэмминга (15;11) 6](#_Toc148303563)

[7. Проверка кода Хэмминга на ошибки 6](#_Toc148303564)

[Задание 5 (89) 6](#_Toc148303565)

[8. Задание 7](#_Toc148303566)

[9. Дополнительное Задание 7](#_Toc148303567)

[Вывод: 8](#_Toc148303568)

[Список литературы: 8](#_Toc148303569)

# Задание:

1. Определить свой вариант задания с помощью номера в ISU (он же номер студенческого билета). Вариантом является комбинация 3-й и 5-й цифр. Т.е. если номер в ISU = 123456, то вариант = 35.
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 отчёте 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 отчёте 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.

# 1. Вариант (91):







# 2-3. Схема кода Хэмминга(7;4)

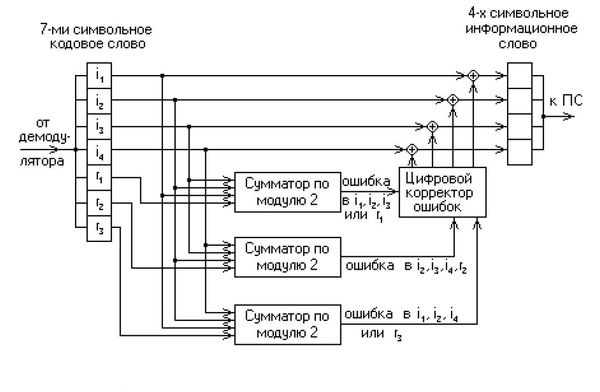


Рисунок 1

# 4.Проверка кода Хэмминга на ошибки

# Задание 1 (75)



R1(1): 0 ⊕ 1 ⊕ 1 = **0**

R2(2): 0 ⊕ 0 ⊕ 1 = **1**

R3(4): 1 ⊕ 0 ⊕ 1 = **0**

**Eror = R3(4)**

Верное сообщение: 

**Ответ:** Ошибка в бите r3, в сообщении 4

# Задание 2 (5)



R1(1): 0 ⊕ 0 ⊕ 0 = **0**

R2(2): 0 ⊕ 0 ⊕ 0 = **0**

R3(4): 0 ⊕ 0 ⊕ 0 = **0**

**Eror = R2(2)** ⊕ **R3(4)** = 2 + 4 = 6

Верное сообщение: 

**Ответ:** Ошибка в бите i3, в сообщении 6

# Задание 3 (47)



R1(1): 0 ⊕ 0 ⊕ 1 = **1**

R2(2): 0 ⊕ 0 ⊕ 0 = **0**

R3(4): 0 ⊕ 0 ⊕ 0 = **0**

**Eror = R1(1)** ⊕ **R2(2)** = 1 + 2 = 3

Верное сообщение: 

**Ответ:** Ошибка в бите i1, в сообщении 3

# Задание 4 (34)



R1(1): 1 ⊕ 0 ⊕ 0 = **1**

R2(2): 1 ⊕ 1 ⊕ 0 = **0**

R3(4): 0 ⊕ 1 ⊕ 0 = **1**

**Eror = R1(1)** ⊕ **R2(2)** ⊕ **R3(4)** = 1 + 2 + 4= 7

Верное сообщение: 

**Ответ:** Ошибка в бите i4, в сообщении 7

# 5 – 6. Схема кода Хэмминга (15;11)

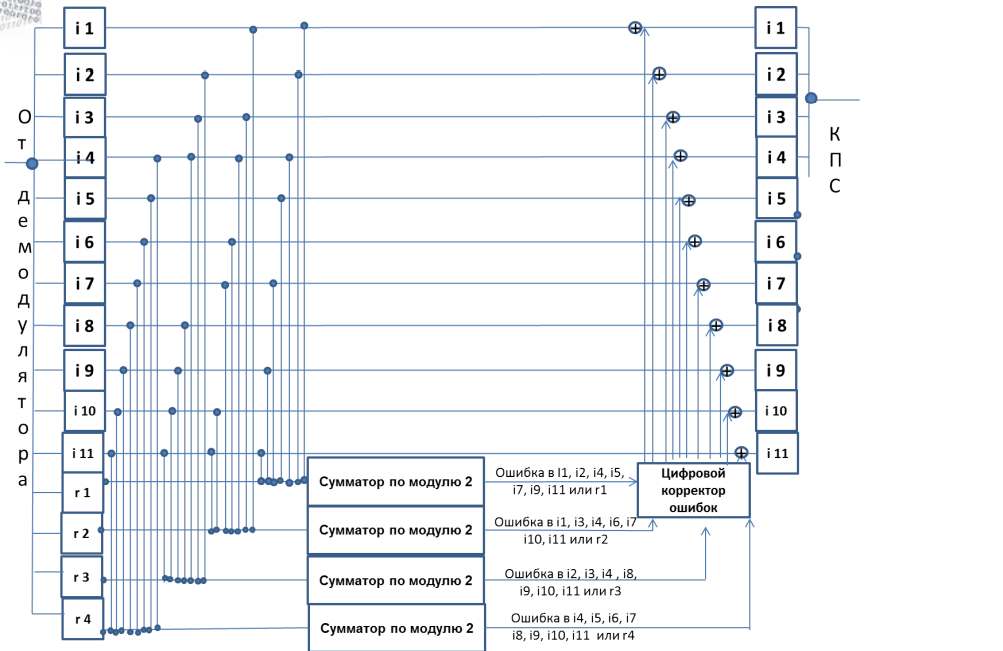


Рисунок 2

# 7. Проверка кода Хэмминга на ошибки

# Задание 5 (89)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 |
| r1 | r2 | i1 | r3 | i2 | i3 | i4 | r4 | i5 | i6 | i7 | i8 | i9 | i10 | i11 |
| 0 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 1 |
| X |  | X |  | X |  | X |  | X |  | X |  | X |  | X |
|  | X | X |  |  | X | X |  |  | X | X |  |  | X | X |
|  |  |  | X | X | X | X |  |  |  |  | X | X | X | X |
|  |  |  |  |  |  |  | X | X | X | X | X | X | X | X |

R1(1): 1 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 1 = **0**

R2(2): 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1 = **0**

R3(4): 1 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 1 = **0**

R4(8): 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 1 = **1**

**Ответ:** Ошибок нет

# 8. Задание

i = (75 + 5 + 47 + 34) \* 4 = 250 \* 4 = 1000

Для определения минимального количества контрольных разрядов воспользуемся формулой:

2r ≥ r + i + 1

2r – r ≥ 1001

Методом подбора получим:

r ≥ 10

Значит min r = 10

Коэффициент избыточности:

k = r / (i + r) = 10 / (1000 + 10) = 0,00990099…

# 9. Дополнительное Задание

# Написать программу на любом языке программирования,

# которая на вход из командной строки 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.

# Проверка ввода. Длина строки = 7, строка состоит ТОЛЬКО из 0 и 1.

**def** validate\_input**(**string**):**

**if** **bool(set(**string**)** **-** **{**'1'**,**'0'**})** **or** **len(**string**)** **!=** 7**:**

**print(**'Неправильный ввод, строка должна быть набором из 7 цифр "0" и "1".'**)**

**exit(**1**)**

# Из строки, полученной на вводе, создаётся список, затем, каждый элемент списка конвертируется в число (int),

# после чего, получается список из чисел.

**def** input\_to\_bits**(**string**):**

**return** **list(map(int,** **list(**string**)))**

# Функция подсчёта и выявления чётных и нечётных сумм для r1, r2 и r3.

# Это необходимо для нахождения значение синдрома, так как синдром

# совпадает с номером позиции искажённого бита.

**def** syndrome**(**arr**):**

s1 **=** **(**arr**[**0**]** **+** arr**[**2**]** **+** arr**[**4**]** **+** arr**[**6**])** **%** 2

s2 **=** **(**arr**[**1**]** **+** arr**[**2**]** **+** arr**[**5**]** **+** arr**[**6**])** **%** 2

s3 **=** **(**arr**[**3**]** **+** arr**[**4**]** **+** arr**[**5**]** **+** arr**[**6**])** **%** 2

**return** **(**s1**,** s2**,** s3**)**

# Проверка на правильность кода Хэмминга,

# Если синдром равен 0, то ошибок в коде нет.

**def** has\_error**(**arr**):**

**return** syndrome**(**arr**)** **!=** **(**0**,** 0**,** 0**)** # True если s1 != 0, s2 != 0, s3 != 0, значит, есть ошибка

# Вывод номера сообщения в котором находится ошибка.

**def** error\_index**(**arr**):**

**return** **int(**''**.**join**(map(str,** syndrome**(**arr**)[::-**1**])),** 2**)**

# Функция, выводит конкретный элемент таблицы, если в нём есть ошибка. (каждому элементу дан свой индекс)

**def** error\_symbol**(**arr**):**

**return** **{** 1**:** 'r1'**,** 2**:** 'r2'**,** 3**:** 'i1'**,** 4**:** 'r3'**,** 5**:** 'i2'**,** 6**:** 'i3'**,** 7**:** 'i4' **}[**error\_index**(**arr**)]**

# Функция выводит список, состоящий из информационных битов i1, i2, i3, i4.

**def** inf\_bits**(**arr**):**

**return** **[**arr**[**2**],** arr**[**4**],** arr**[**5**],** arr**[**6**]]**

# Функция финального вывода, необходимая для fixed\_message(arr)

**def** make\_result\_message**(**bits**):**

**return** ''**.**join**(map(str,** bits**))**

# Функция возвращающая правильное значение для информационных битов i1, i2, i3, i4

**def** fixed\_message**(**arr**):**

**if** **not** has\_error**(**arr**)** **or** error\_symbol**(**arr**)[**0**]** **==** 'r'**:**

**return** make\_result\_message**(**inf\_bits**(**arr**))**

ind **=** **int(**error\_symbol**(**arr**)[**1**])** **-** 1

result **=** inf\_bits**(**arr**)**

result**[**ind**]** **=** **(**result**[**ind**]** **+** 1**)** **%** 2

**return** make\_result\_message**(**result**)**

inp **=** **input(**'Введите набор из 7 цифр «0» и «1», записанных подряд: '**)**

validate\_input**(**inp**)**

bits **=** input\_to\_bits**(**inp**)**

**if** has\_error**(**bits**):**

**print(**f'> В сообщении ошибка!\nОшибка в символе {error\_symbol**(**bits**)**}'**)**

**else:**

**print(**'> В сообщении нет ошибок!'**)**

**print(**f'Правильное сообщение: {fixed\_message**(**bits**)**}'**)**

# Вывод:

Я разобрался в том, для чего нужен и как работает код Хэмминга и попрактиковался в написании кода на языке python

# Список литературы:

1. Основы цифровой радиосвязи. Помехоустойчивое кодирование: метод. указания / сост. Д. В. Пьянзин. – Саранск : Изд-во Мордов. ун-та, 2009. – 16 с.
2. Коды и устройства помехоустойчивого кодирования информации / сост. Королев А.И. – Мн.: , 2002. – с.286